Mechanical Trading Systems

earik beann

W59 | WAVE59® TECHNOLOGIES

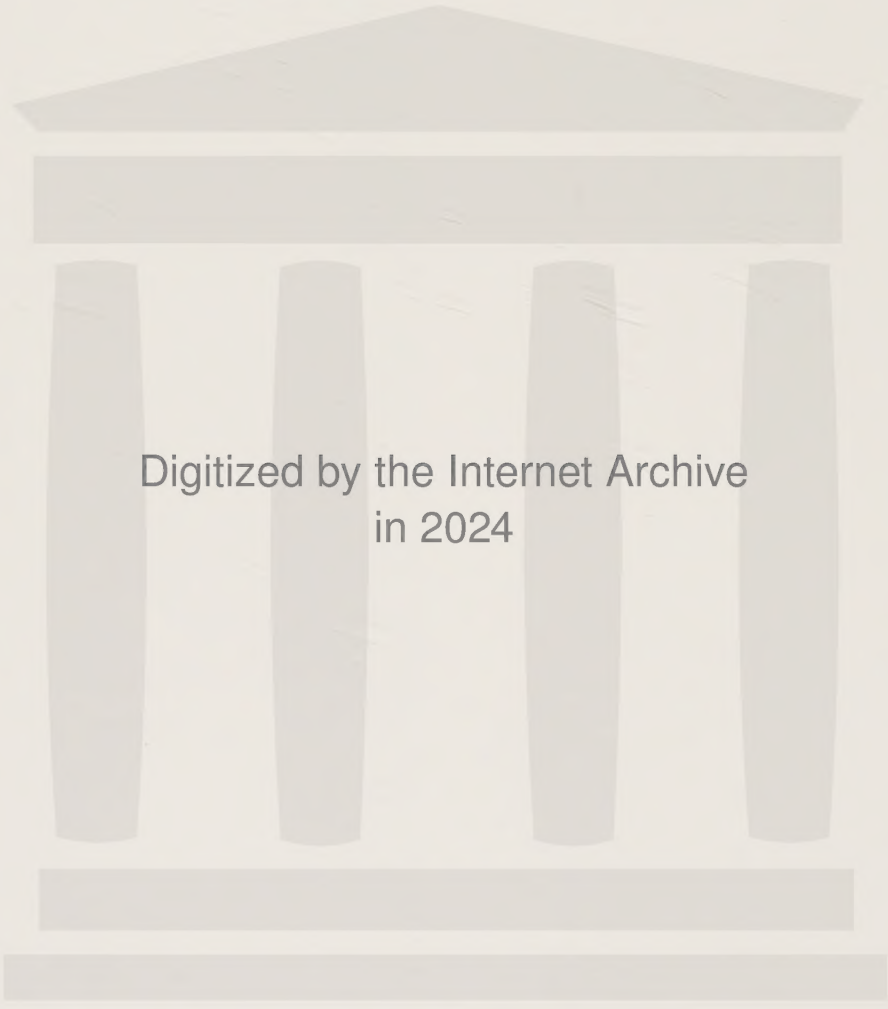
You now hold in your hands the biggest, most turn-key manual Wave59® has ever published.

Mechanical trading systems are one of the greatest developments in the history of trading. Turn them on, connect them to your broker, and they will trade for you. However, ...

developing a good mechanical system is not always so easy, especially if you aren't aware of the many pitfalls involved. Although Earik is best known for some of his discretionary approaches, he got his start building automated trading systems long before Wave59 even existed. From his days trading Treasury Bonds systems at the CBOT, through his work setting up a private fund to trade the ES, he has over twenty years of testing, tweaking, and innovating mechanical systems to trade financial markets.

This course is all about mechanical trading systems: how to evaluate them, how to build them, and how to trade them. But in typical Wave59 fashion, we're going to go about doing that a little differently. Unlike every other systems book in existence, you'll learn by taking apart actual, working systems that Earik has used in live markets throughout the years. These are not dumbed-down, for-example-only systems, but real life, actual trading methods that you can take today and implement in your own trading. Everything is completely revealed, including QScript® source code, so if you want to jump to the end and start trading the systems, go for it — everything you need to implement these methods can be found in the appendix.

Copyright © 2013 by Wave59 Technologies Int'l, Inc. All rights reserved.
This book may not be copied in part or full
without express written permission from the publisher.


Digitized by the Internet Archive
in 2024

https://archive.org/details/mechanicaltradin0000eari

Mechanical Trading Systems

By Earik Beann

Copyright © 2013 by Wave59 Technologies Int'l, Inc. All rights reserved.

A friendly reminder from our lawyers and various governmental agencies:

U.S. Government Required Disclaimer - Commodity Futures Trading Commission Futures and Options trading has large potential rewards, but also large potential risk. You must be aware of the risks and be willing to accept them in order to invest in the futures and options markets. Don't trade with money you can't afford to lose. This is neither a solicitation nor an offer to Buy/Sell futures, stocks or options on the same. No representation is being made that any account will or is likely to achieve profits or losses similar to those discussed in this book. The past performance of any trading system or methodology is not necessarily indicative of future results.

CFTC RULE 4.41 - HYPOTHETICAL OR SIMULATED PERFORMANCE RESULTS HAVE CERTAIN LIMITATIONS. UNLIKE AN ACTUAL PERFORMANCE RECORD, SIMULATED RESULTS DO NOT REPRESENT ACTUAL TRADING. ALSO, SINCE THE TRADES HAVE NOT BEEN EXECUTED, THE RESULTS MAY HAVE UNDER-OR-OVER COMPENSATED FOR THE IMPACT, IF ANY, OF CERTAIN MARKET FACTORS, SUCH AS LACK OF LIQUIDITY. SIMULATED TRADING PROGRAMS IN GENERAL ARE ALSO SUBJECT TO THE FACT THAT THEY ARE DESIGNED WITH THE BENEFIT OF HINDSIGHT. NO REPRESENTATION IS BEING MADE THAT ANY ACCOUNT WILL OR IS LIKELY TO ACHIEVE PROFIT OR LOSSES SIMILAR TO THOSE SHOWN.

THIS BOOK IS INTENDED FOR INFORMATIONAL AND ENTERTAINMENT PURPOSES ONLY AND SHOULD NOT BE RELIED ON AS INVESTMENT, FINANCIAL, OR LEGAL ADVICE. USER ACKNOWLEDGES THAT WAVE59 TECHNOLOGIES, ANY AFFILIATES, SUBSIDIARIES, EXCHANGES, DATA VENDORS OR RELATED THIRD PARTIES (COLLECTIVELY, "WAVE59") DO NOT GUARANTY THE ACCURACY OR COMPLETENESS OF THE CONTENTS HEREIN. WAVE59 DOES NOT GUARANTEE RESULTS, FINANCIAL GAIN, OR ANY OTHER REMUNERATION OR BENEFIT. CONTENT SHOULD NOT BE RELIED ON AS THE BASIS FOR LEGAL, PERSONAL, BUSINESS, FINANCIAL, RELATIONSHIP OR ANY OTHER DECISIONS. WAVE59 IS NOT PROVIDING ANY RECOMMENDATIONS OR ENDORSEMENTS FOR ANY INVESTMENT OR RELATED INFORMATION.

Table of Contents

1. Introduction 1

Programming Requirements 3

Available Indicators 3

Backtesting 4

Position Sizing Applications 4

Emotional Requirements 4

Time Requirements 5

2. The System Report Explained 9

Summary 10

Total Profit 11

Long Profit 11

Short Profit 11

Accuracy 12

Winning Trades 12

Losing Trades 12

Total Trades 12

Average Win 12

Average Loss 12

Average Trade 13

Win/Loss Ratio 14

Largest Win 15

Largest Loss 16

Longest Winning Streak 16

Longest Losing Streak 16

Maximum Shares/Contracts Held 16

Gross Profit 17
Gross Loss 17
Profit Factor 17
Annual Return 17
Max Drawdown 17
Max Drawdown Percent 21
Max Drawdown Date 21
Max Drawdown Percent Date 21
Longest Time Underwater 21
Longest Time Underwater Date 22
Trade by Trade 23
Equity Curve 24
Annual Results 31
Monthly Results 32
3. William Tell Bonds 35
4. The Fallacy of Optimization 70
5. Mean Reversion Systems 86
6. Stop Losses, Targets and Other Ways to Lose Money 97
7. Lunar System 111
8. Machine Learning 147
Hive Technology 148
Genetic Algorithms 156
Good Inputs 174
Good Fitness Function 175
Reasonable Expectations 175
9. Random Numbers 177

10. Merged Systems 191
11. Position Sizing 203
12. Options 220
13. Conclusion 247
14. Appendix 252
Earik_System_WTBonds 255
Earik_System_AdaptiveCrossoverFade 264
Earik_SimpleAvgCrossover 271
Earik_System_BCI 272
Earik_System_BookSphericalTester 273
PUC_Lunar_Reversal 275
PUC_LunarSignal 278
PUC_Lunar_NthProfitableClose 280
PUC_Lunar_MultipleCt_ProfitableCloseExit 283
Hive_Chicago 286
Hive_Chicago 287
Hive_Chicago2 295
Earik_System_UltramomInv 296
Earik_System_BookGA1 297
Earik_System_BookGA2 298
Earik_System_RandomTest 299
Earik_System_RandomTradeAll 301
R11 303
Earik_System_RandomOneContract 306
Sign 307
Earik_System_Merged 308

Earik_System_MergedVoting 310
Earik_System_SimplePosSizing 311
Earik_System_ModifiedPosSizing 312
Earik_System_StockPosSize 314
Autotrade_Template 315

Table of Figures

Figure 2.1: Example System Report - Summary 10
Figure 2.2: Example System Report - Trade by Trade 23
Figure 2.3: Example System Report - Equity Curve 24
Figure 2.4: Example System Report - Equity Curve without Initial Equity 25
Figure 2.5: Ideal Equity Curve 26
Figure 2.6: Bad Equity Curve = BAD system 28
Figure 2.7: An Equity Curve of a System with No True Edge 29
Figure 2.8: Equity Curve that Shows Some Promise 30
Figure 2.9: Example System Report - Annual Results 32
Figure 2.10: Example System Report - Monthly Results 33
Figure 3.1: William Tell Bonds Pattern #14 39
Figure 3.2: William Tell Bonds Pattern #14 System Report 40
Figure 3.3: William Tell Bonds Pattern #14 Equity Curve 44
Figure 3.4: William Tell Bonds Pattern #14 Monthly Results 45
Figure 3.5: William Tell Bonds Pattern #2 46
Figure 3.6: William Tell Bonds Pattern #2 System Report 47
Figure 3.7: William Tell Bonds Pattern #2 Equity Curve 48
Figure 3.8: William Tell Bonds Pattern #12 50
Figure 3.9: William Tell Bonds Pattern #12 System Report 51
Figure 3.10: William Tell Bonds Pattern #12 Equity Curve 52
Figure 3.11: William Tell Bonds Pattern #16 54
Figure 3.12: William Tell Bonds Pattern #16 Equity Curve 55
Figure 3.13: William Tell Bonds Pattern #18 Equity Curve 56
Figure 3.14: William Tell Bonds Aggregate System Equity Curve 60
Figure 3.15: William Tell Bonds Aggregate System Equity Curve Lifetime 61
Figure 3.16: William Tell Bonds First Client Loss After Live Trading 66
Figure 3.17: William Tell Bonds First Client Loss After Live Trading (zoomed in) 67
Figure 4.1: System Report for Optimized Moving Average System 74
Figure 4.2: Equity Curve for the MA System Optimized for 1995-2004 75
Figure 4.3: Out of Sample Execution (2005) of MA System Optimized for 1995-2004 76
Figure 4.4: Out of Sample Execution (2006) of MA System Optimized for 1996-2005 77
Table 4.1: P&L for Optimization Values over 10 Year Periods for MA System 78

Table 4.2: P&L for Optimization Values over Single Year Period for MA System 80
Figure 4.5: Adaptive System (main logic) 83
Table 4.3: Adaptive System Optimization 84
Figure 5.1: Balanced Center Line (BCI) 87
Figure 5.2: BCI System Report - S&P 500 Futures 89
Figure 5.3: BCI Equity Curve - S&P 500 Futures 90
Figure 5.4: BCI Annual Results - S&P 500 Futures 91
Figure 5.5: BCI System Report - SPY 93
Figure 5.6: BCI Equity Curve - SPY 94
Figure 6.1: BCI Stop Loss Optimization Test 100
Figure 6.2: BCI Example Outcome from Having No Stop 102
Figure 6.3: BCI Profit Target Optimization Test 105
Figure 6.4: BCI SL and PT Optimization Run Worst 20 108
Figure 6.5: BCI SL and PT Optimization Run Best 20 109
Figure 7.1: Ephemeris within Wave59 112
Figure 7.2: Trine Aspect Sat-Nep 240 113
Figure 7.3: Declination of Planets within Ephemeris 115
Figure 7.4: Square Mars-Saturn 116
Figure 7.5: Simple Venus-Sun Aspect System 118
Figure 7.6: Sun-Venus Aspect System Equity Curve 119
Figure 7.7: Sun-Venus Aspect Threshold Levels Optimization 120
Figure 7.8: Lunar Ratio System - 15 day Hold Optimization 121
Figure 7.9: Moon-Venus Ratio System Equity Curve 122
Figure 7.10: Plotting the Ratio as an Indicator 123
Figure 7.11: Lunar System Buy Line Optimization 125
Figure 7.12: Lunar System Sell Lines Optimization 127
Figure 7.13: Lunar System with Optimal Signal Lines 130
Figure 7.14: Lunar System with Optimal Signal Lines Equity Curve 131
Figure 7.15: Dow Jones Big Trends 132
Figure 7.16: Lunar System with Short Filter System Report 134
Figure 7.17: Lunar System with Short Filter Equity Curve 135
Figure 7.18: Lunar System Nth-Profitable Close Exit Optimization 136
Figure 7.19: Lunar System 2nd Profitable Close System Report 138
Figure 7.20: Lunar System 2nd Profitable Close Equity Curve 139
Table 7.1: Original Lunar vs. 2nd Profitable Close Dollars Earned per Unit of Risk 140

Figure 7.21: Lunar System Combined Exit Results System Test 141
Figure 7.22: Lunar System Combined Exit Annual Results 142
Figure 7.23: System Combined Exit Equity Curve 143
Figure 8.1: Chicago Hive Chart Indicator 150
Figure 8.2: Chicago Hive System Report 151
Figure 8.3: Chicago Hive Equity Curve 152
Figure 8.4: Chicago Hive Modified System Report 154
Figure 8.5: Chicago Hive Modified Equity Curve 155
Table 8.1: Genetic Algorithm Example Generation 1 158
Table 8.2: Genetic Algorithm Example Generation 1 Fitness 158
Table 8.3: Breakdown of Crossover at Gene 5 159
Table 8.4: Genetic Algorithm Example Generation 2 160
Table 8.5: Genetic Algorithm Example Generation 3 161
Figure 8.6: Inverse Momentum System Report 164
Figure 8.7: Inverse Momentum Equity Curve 165
Figure 8.8: Inverse Momentum Look Back Length Optimization 166
Figure 8.9: Genetic Algorithm Built System #1 System Report 168
Figure 8.10: Genetic Algorithm Built System #1 Equity Curve 169
Figure 8.11: Genetic Algorithm Built System #2 System Report 171
Figure 8.12: Genetic Algorithm Built System #2 Equity Curve 172
Table 9.1: Trade Statistics for Random Entry Test 178
Figure 9.1: Random System with Buy/Sell on Each Bar 180
Figure 9.2: Random System Buy/Sell on Each Bar Exit Optimization - Best 181
Figure 9.3: Random System Buy/Sell on Each Bar Exit Optimization - Worst 182
Figure 9.4: Random System Buy/Sell on Each Bar - Best Parameter Set Equity Curve 183
Figure 9.5: Random System Buy/Sell on Each Bar - Best Parameter Set System Report 184
Figure 9.6: Random System Buy/Sell on Each Bar - Defined as an Indicator 185
Figure 9.7: Random System Buy/Sell Indicator - Exit Optimization 186
Figure 9.8: Random System Buy/Sell Indicator System Report 187
Figure 9.9: Random System Buy/Sell Indicator Equity Curve 189
Table 10.1: List of Systems Available for Trading 193
Figure 10.1: Simultaneous Merged System Report 195
Table 10.2: Simultaneous Merged Equity Curve 196
Table 10.3: Voting System Report 198
Table 10.4: Voting Equity Curve 199

Table 10.5: System Results Comparison 200
Table 11.1: Voting System Report 205
Table 11.2: Voting System with Position Sizing System Report 207
Table 11.3: Voting System with Position Sizing Equity Curve 208
Table 11.4: Voting System with Enhanced Position Sizing System Report 212
Table 11.5: Voting System with Aggressive Position Sizing System Report 215
Table 11.6: Voting System with Aggressive Position Sizing Equity Curve 216
Table 11.7: Voting System with Ultra Aggressive Position Sizing System Report 218
Figure 12.1: Voting System Trading a Single Share of SPY 221
Figure 12.2: Voting System Trading the Maximum Theoretical Size for SPY ($100k) 223
Figure 12.3: Voting System Trading the Maximum Theoretical Size for SPY ($5k) 225
Figure 12.4: Voting System Max Theoretical Size for SPY ($5k) Equity Curve 226
Figure 12.5: Equity Curve Trading SPY 229
Figure 12.6: Equity Curve for Option Strategies 231
Figure 12.7: ATM Options Sold vs. Stock Equity Curve 232
Figure 12.8: Option Buying vs. Stock Equity Curve 233
Table 12.1: Stock Equivalency for Option Strategies 234
Table 12.2: Average Option Price for Option Strategies 234
Figure 12.9: Equivalency of 4.5 * OTM4 vs. Stock Equity Curve 235
Figure 12.10: Stock Equity Curve Trading the Max # of Shares Using 50% Margin 236
Figure 12.11: OTM4 @ 5% Risk vs. Stock Equity Curve 237
Figure 12.12: OTM4 @ 20% Risk vs. Stock Equity Curve 238
Figure 12.13: OTM4 @ 50% Risk vs. Stock Equity Curve 239
Figure 12.14: OTM4 Price Adj. Position Sizing (multiplier 0.5) vs. Stock Equity Curve 242
Figure 12.15: OTM4 Price Adj. Position Sizing (multiplier 0.75) vs. Stock Equity Curve 243
Figure 12.16: Proxy Results for the OTM4 with Price Adj. Position Sizing 244

1. Introduction

Welcome to the Mechanical Trading Systems book. This manual will be a little different from some of the other ones I've written, because it deals with a subject that hasn't been a primary focus at Wave59 (at least not publicly) very often. Yes, Wave59 does offer system testing, as well as some very unique artificial intelligence, but for the most part Wave59 has been perceived as primarily a tool for discretionary trading, and not systematic trading.

There are really two major categories that traders find themselves in when implementing any sort of approach designed to take profits out of the markets. They can either be discretionary traders, or systematic ones. A **discretionary** trader is one that makes the ultimate trading decision themself, based on some collection of techniques that they believe gives them an edge. This includes old school technical analysts with their hand-drawn charts, modern indicator traders watching squiggly lines on a computer screen, as well as really long term fundamental traders. These traders take in information, and then use that information to make a decision about markets, which is then acted upon by manually placing trades with a broker. The key point here is that these traders are part of the equation, and if you take the human out of the process, no trades will be generated.

The second category of traders are **systematic** traders. These traders develop a set of rules that lead to completely automatic trading decisions. Various inputs go in to the model, a flow chart of some sort is followed, and the final output is a trading decision. The key definition for systematic traders is that their systems require no human involvement at all, and can be automated all the way up to the point of placing the order itself. So although a human creates the system (usually!), once it has been finalized, it no longer requires any human interaction and the system handles all aspects of running the trading operations itself, including trade selection, stop-loss management, profit taking, position sizing, etc.

Both of these approaches work, and there are traders in both camps that have been extremely successful in the markets. However, their methods will tend to differ, and those differences become more and more pronounced as you go deeper and deeper into each respective field.

As discretionary trading requires the inclusion of a human in the analysis component, the advantages of the human brain can be harnessed. This gives the discretionary traders access to various forms of analysis that are extremely difficult, if not impossible, for most systematic traders to implement, due to the technical limitations. The most obvious example of this type of analysis is geometric pattern matching, where a chart is scaled in a certain way, and various shapes and patterns are applied to the chart to take advantage of the geometric nature of markets. Spirals, squares, flower patterns, and all sorts of other shapes can be applied to markets to generate very potent timing information about change-in-trend dates in the future. The human brain is a fantastic pattern matcher, and is extremely well suited to this form of analysis. Humans can find pictures in clouds with no trouble, and that same ability serves us very well when using this type of approach in markets.

Computers, on the other hand, are terrible at finding visual patterns, and the most advanced computers in the world are still nowhere close to even the most knuckleheaded human when it comes to looking for shapes in the chart. Because of that, you'll tend not to see a whole lot of geometric pattern matching in the systems world, although discretionary traders will have no problems implementing that sort of thing.

On the flip side, computers are very good at seeing other kinds of patterns, such as numerical relationships between multiple different data sets, and they can perform thousands of comparisons and calculations in a blink of an eye. Speed, in particular, is the forte of the computerized trader, and this gives computers the ability to trade at speeds and time frames that humans cannot possibly match. So while a human might have the ability to watch one pattern on the market they are trading, a computer can watch for patterns between one market and another, or between one market and *ALL* the other markets.

Because systematic traders require computers to implement their systems, you'll find that systems are based more on technical indicators, and numerical representations of markets, rather than visual representations of markets. This will become clear as you start working through this book, especially when it comes to charts.

As mentioned earlier, you can be successful both in the discretionary and systematic realms, so one approach is not necessarily better than the other. The differences

between them tend to do more with the temperament of the trader involved than anything else, and systematic-oriented traders will tend to gravitate towards systems, while the discretionary traders will gravitate towards discretionary methods. But there are definitely some pros and cons to each approach, and it is helpful to know the strengths and weaknesses of both types of trading, especially when first getting one's feet wet in the trading world. Let's run through a quick comparison of these methods so we can compare and contrast them.

Programming Requirements

Mechanical Systems are rule-based, which means they have to be programmed into a computer using either a full-blown programming language like C++, or a trading-system scripting language like Wave59's QScript®. This means that if you are going to develop your own systems, you need to come to grips with doing a bit of coding. Don't be afraid of this step, it's not that bad, but it definitely can be intimidating at first. It is important to realize that if you don't learn a little bit of programming, the only way you'll be able to trade systems is if you buy them from someone else, which puts you at the mercy of the marketplace. This facet of the industry tends to be full of very hyped up methods that don't actually work as advertised. Of course, discretionary traders have this very same problem!

Available Indicators

Discretionary traders definitely have an advantage here. Not only can they use all the same indicators that system traders use, they also have access to more subjective approaches that defy being programmed. Elliot Wave analysis is a good example of this. It's a very interesting technique, but it's subjective, which means that you can try to program it into software (and there are tons of people who have attempted this), but will probably never get it quite right. So discretionary traders can easily slap their five waves up and three waves down and make some trades based on this information, but systematic traders have a much harder time of it. Having something like the *Unified Theory of Markets* behind you can really put the odds in your favor as a discretionary trader, and it's a shame when all that power can't be used inside a mechanical system due to programming difficulties. This brings up a subtle, but important point. There are limitations to what can go into a system, and if an indicator can't be automated, then it can't be used in a mechanical system.

Backtesting

Backtesting is the process where a method is run using past data to see how well it would have performed during the period in question. Due to the nature of mechanical systems, this is an area where they truly shine. Once a system has been automated, it is a simple process to run it on past history to collect all sorts of information about its performance. There are plenty of software programs available to help with this process, and you can run a method on 100 years of data and get results in a few seconds. Those discretionary traders attempting anything close to this will spend days, weeks, or even months struggling with an incredibly tedious, error-prone task. This is where the system traders start winning some serious points. They might not be able to use all the same approaches as the discretionary crowd, but they can quickly and easily test all sorts of different ideas, and learn what works and what doesn't. If you're going to be profitable in this business, you need to test your methods, and having a computer do it for you is much easier than doing it by hand, which is the only option for the discretionary traders.

Position Sizing Applications

When placing a trade, you need to know not only whether to buy or sell, but also how much to buy or sell. **Position sizing** refers to various methods of dealing with the size question in the most efficient way. This tends not to be nearly as popular a topic as dealing with finding entries, but simple tweaks here can turn average systems into fantastic ones. As traders, our ultimate goal is to grow our account as quickly as possible while staying within our risk constraints. Position sizing is a way to turn a regular, smooth equity curve into an explosive, exponential one that climbs straight through the roof. If you are interested in eventually making millions of dollars a year off your systems, it's the position sizing aspect of trading that will take you there. Although discretionary traders can do better than systematic ones trading fixed size, I don't know any discretionary traders that can outperform a good system with aggressive position sizing.

Emotional Requirements

One of the nice things about mechanical systems is that they take all the stress out of placing trades. Unlike discretionary trading, where a trader is always second guessing themselves at some point during the trade, a systematic trader is able to mostly just sit back and watch the trade unfold. Although the fear/greed cycle is still present, the

system itself acts as a crutch against making poor decisions in the heat of the moment. Just follow the system, and things work out. There is no new information that can change a trade, and there are no actions to take during a trade. As systems remove the trader from the process, the trader is not required to go through the same pressures during the trading process as they would were they involved. This is not to say that system traders don't have their freak-out moments, only that they tend to be more contained. It's a huge advantage.

Time Requirements

As many of you know, I'm not as involved in the software side of Wave59 Technologies anymore, and have been pursuing other things during the day rather than programming software. After I wrote *The Unified Theory of Markets* book, I went through a period of very active day trading, on fast time frames like 2-minute charts. The first few weeks were really exciting, and reminded me of my early 20's, back when I worked at the Chicago Board of Trade. I'd come in first thing, get a feel for the market, and would then watch things open up and would be duking it out all day long until the close. There is something very pure about going one-on-one with the market, working things out, and being rewarded (or not) based on nothing more than your own skill and approach. This is what got me interested in trading in the first place, and is why so many others are attracted to this business.

What a lot of non-traders don't realize though, is that once you get past the initial rush of day trading full time, it becomes rather tedious. The fact of the matter is that you become tied to your computer screen, and find it difficult to leave. This is especially true if you are waiting for one or two big turns in the day – if you miss those turns, you miss the whole day, and all that work and analysis you did goes down the drain. So despite the huge advantages of trading for oneself, what can happen is that trading becomes a job, and it can be a drag. It can also be stressful, especially if you lost money. Even worse than losing trades is missing good ones, due to being distracted, or for other reasons not related to actual market analysis. You can't just coast along, but really have to be on top of your game, or the market will eat your lunch.

At the same time as I was day trading, I was also running a mechanical system that I had developed for a private fund. As a member of the fund, I've invested some of my own money, and for years we've been running this approach very profitably. During the same

period of time that I had been day trading, this fund had been seriously kicking the market's butt, and had generated some very attractive returns. The percentage gains weren't close to what I had seen with the *Big Bertha/UTM* material, but in dollar terms the fund was way ahead of my private trading due to its size, and the fact that it was implementing some very aggressive position sizing. Again, a good mechanical system with dynamic position sizing will outperform a great discretionary method with fixed size, especially when looked at over time.

So, like all good traders, after a few months of running my trading operations, I sat down and reviewed my trading journal, and took stock of my situation. On one hand, I had the results from my day trading operations, and on the other I had my results from the fund. How had the two done in comparison to each other? The fund had out-performed my own trading on a dollar basis by a significant factor. So, judging by actual money-in-the-bank numbers, I had made more on my fund than I had on my trading operations. How much time had I put into the day trading operations? I had worked all day long, 5 days a week, plus having to get up early, and sometimes working late. How much time had I spent working on the fund? Less than 1 minute per day.

I like markets, and I like figuring things out, but I'm no glutton for punishment, and I value my time. That was a huge slap in the face to me. Rather than fooling around watching markets, I could have been sleeping in, exercising, going out to lunch, playing games... heck, doing anything. Just let the money roll in and be done with it. It's hard to argue that once you see it up close, and after that point I had some trouble really justifying the day trading. If that's all I had going for me, no problem, I'd do it, but with a successful mechanical system, it becomes a pretty hard sell.

This, above all else, is the massive advantage mechanical systems bring to the table. Once you've built a good one that has been tested and that you trust, you can send it out into the world and it will go make money for you. There are not many other professions where you can earn money without doing any work, but this is one of them.

So after a brief comparison, you can see some of the benefits of running mechanical systems, and that brings us to the purpose of this book. In the final chapter of *The Unified Theory of Markets*, I had wondered whether or not that would be the last trading book I'd ever write due to the impossibility of being able to top that material. At this point, that statement still stands, in the sense that I don't have anything else to

offer above and beyond what I've already put out there for the discretionary traders. However, I do have quite a bit of material to offer the systematic crowd. So rather than trying to outdo the *UTM* book, let's take a fork in the road and explore a completely different path in the trading landscape. Since I don't have to day trade anymore, I do have some time on my hands to write. ☺

The purpose of this book is threefold. First, I'd like to bring the Wave59 community members (and all my fans in the pirate world!) up to speed as far as how to judge a mechanical system. There are a lot of programs that can generate system reports, but until you know how to read one of those reports, you really have no way of telling whether or not the system that you just tested has any promise. Evaluating systems is actually quite simple once you understand how to do it, and that knowledge can save you from some seriously costly mistakes, the least of which being the sinking of a large chunk of hard-earned cash into one of the poorly designed $3000 trading systems you'll find advertised in the back of some of those trading magazines.

Rather than being pedantic and speaking in terms of math and formulas for the rest of the book, we're going to approach this task in a more interesting way, by actually looking at a number of the systems I've come up with over the years. The best way to learn is by doing, and so we'll walk through each of these systems with the same mindset as if we were developing them. As we look at the good and bad points of each method, we'll reference the system report to provide clarity as far as what all these metrics really mean practically. You'll also be able to follow my own development over the years in terms of what I was looking for in a trading system, which I would expect to be a very close mirror to what the average system trader goes through over a long period of time. Hopefully, you'll be able to learn from my mistakes and won't have to experience some of the same lessons yourself.

The second purpose to this book is to provide you with some trading systems that you can take and use in your trading today. I remember back when I first got started trading, one of the big purchases I made was a full license to the industry standard (as they called themselves) trading system development and testing platform. You probably know who I'm talking about. In their drop down menu, they had a Trading System tab, where you could apply various pre-built mechanical systems to your market. I remember being super excited about that, especially when I applied the very first system to my chart, and saw all those buy and sell arrows pop up. Unfortunately, my

excitement was short lived, as when I pulled up the system report, I found that the system performed terribly, losing money every year at a consistent rate. Every single one of the systems provided in that platform was a consistent loser, across all markets and time frames. Yes, moving averages can be made to generate mechanical systems from which you can place trades. But no, you're not actually going to make any money using any of them. And so my fancy "trading system" software actually didn't come with any real trading systems!

With that in mind, unlike all the other systems manuals I've ever seen, we're not going to be looking at stupid elementary school trading systems that are only there for illustrative purposes. These are all actual systems that I've traded in my own account. Granted, they all haven't stood the test of time, as some of them were over-optimized and built when I didn't really know quite what I was doing. I'm going to discuss those to show you what not to do. However, I'm also going to share a few trading systems that I believe are extremely robust, which can be applied to your own trading account today. So if you want to skip over the theory and just trade, you can do that too. The source code is all here – just copy and paste it into Wave59, and go for it.

Finally, I hope that by learning to evaluate systems properly, and by walking through some real trading systems, you'll get inspired to create your own custom systems based on your own ideas about markets. The entire community of traders benefits from one or two people sharing their ideas and teaching others, and hopefully this manual will light a fire under a few of those individuals and get them working on the next generation of tools.

Let's get started!

2. The System Report Explained

When you apply a trading system to a chart in any of the modern trading software programs, you'll receive a system report which contains a list of various metrics that tell you how well your system did when trading over a specific period of time. The metrics used can vary by software program, but there are some core metrics that you'll find in all of them. It's good to have all of this in one place for future reference, so in this chapter we'll go through one of these reports and familiarize ourselves with what each metric means.

We will start with the Summary tab from Wave59's system reporting module, and make our way through the entire report in order.

Summary

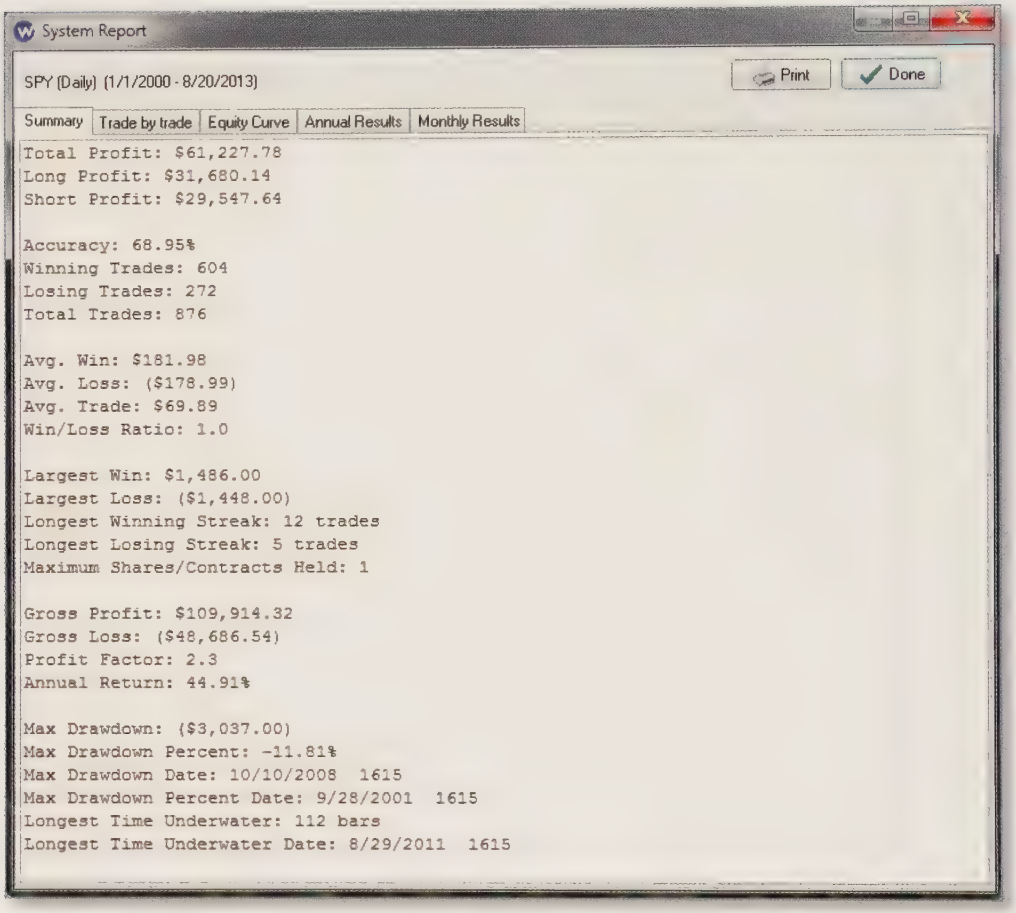


Figure 2.1: Example System Report - Summary

The figure above shows a system report from Wave59 running a system that I'm currently trading on the SPY, a popular ETF traded on the AMEX. As we know, a mechanical system follows a set of rules to automatically generate buy and sell signals for a particular market. As everything is done automatically, we can apply our system to a historical data set to see how the system would have done had we traded this strategy in the past. This is done to prove out the system logic, as well as determine whether or not the approach is worth using in real time.

As you can see from our report, there are a number of metrics listed in the "Summary" tab of this report. Let's briefly touch on each one, working from the top down.

Total Profit

This metric tells you how much money the system gained or lost over the test period. In our example, we traded 100 shares of SPY starting January 1, 2000 all the way up through August 20, 2013. During this time, our account grew in value by $61,227. This is a very easy metric to understand. You want it to be positive, and you want it to be large.

Of course, just because you made money during a specific period of time doesn't mean that your system can still be traded. How you make the money is actually more important than how much was made in the past, and as we work through these numbers we'll start to get a much better feel for that. The key point about this metric is that is really needs to be positive. A successful backtest doesn't guarantee that the system will work in the future, but a failed backtest almost certainly does guarantee that the system will not work moving forward.

Long Profit

This metric is very similar to Total Profit, except it only shows how much money the method made when trading the long (buy) side of the market. One of the things to look at is to see how much of the overall profit came from the long side compared to how much of the profit came from the short side. Ideally, you'd like to see a nice balance between the two. If it turns out that all the profit comes from only going long, it tells you that the system can't handle the conditions that develop during bear markets. Alternatively, it might tell you that your market behaves differently when going up compared to going down, and you might need to treat those two conditions in different ways. We'll see an example of this when we look at the Lunar System later on.

Short Profit

Like the previous metric, this value shows how much money was made when only considering the short (sell) side. Some systems don't go short, and will only be long or flat. In those cases, this value will be zero.

Accuracy

This tells you what percentage of all trades taken resulted in a profit. Higher numbers are obviously more desirable. This particular metric tends to be treated as being more important than it really is, especially by beginning traders. We'll circle back around to this one in a lot more detail in the next chapter when we discuss the William Tell Bonds system.

Winning Trades

This is the number of winning trades that the system took during testing. It's not terribly useful on its own, but gets used in other metrics such as the accuracy number.

Losing Trades

This is the number of losing trades that the system took during the backtest period. It's basically the opposite of the last metric.

Total Trades

This is the total number of trades taken during the backtest run, including both winners and losers. Ideally, you want to see a system take lots of trades in a test. The fewer trades your system takes, the more cautious you need to be when evaluating its results. Conversely, the more times your system enters the market, the more you can count on its results as being significantly valid from a statistical point of view. As a rule of thumb, 30 trades is about the smallest number of trades you need in order to be able to trust your backtesting results. Although I have traded patterns with only 30 occurrences in the past (we'll discuss that system later), these days I only work with methods that have hundreds, and sometimes thousands, of hypothetical trades to base my assumptions off of. The point of backtesting is to pin down what sort of an edge you've got, and you can't do that with certainty unless you can see your system in action a whole bunch of times.

Average Win

This is the amount of money made, on average, each time a winning trade is taken.

Average Loss

This is the amount of money lost, on average, each time a losing trade is taken.

Average Trade

This is one of the more important metrics. To compute it, you take the total profit generated, and divide by the total number of trades taken. The result is how much you can expect the system to generate, on average, every single time you place a trade. This represents the edge your system has in the market. When looking at the average trade value, the key question to ask yourself is whether this value is enough to overcome the costs of trading, specifically commissions, slippage, and errors.

If you are trading an intraday ES system and placing market orders, you can count on the fact that you'll lose one tick on the entry and one tick on the exit, at a minimum. On top of that, you're going to be paying a commission, and on top of all that, if you've got a human involved in the process in any way, shape, or form, eventually that human will do something stupid and cost the system some profits. So you have to make sure that the system makes enough money to absorb all of this and still come out ahead. If your average trade is only $30, you will definitely be struggling in this situation. This is one of the reasons why it's easier to develop daily systems rather than intraday ones – the average trade tends to be much higher.

Take a look at the example system we are looking at – it has an average trade value of $69. This is a stock trading system, and slippage values on this particular market are around one cent per side. Commissions are a fraction of a cent per share, so given a 100-lot, your costs will be a few dollars. Since we have $69 to work with, we have a ton of room left over to generate profits in the account.

Each market and time frame will have a slightly different profile, so you have to judge this number in the context of your particular market. $69 is a high average trade value for stocks, but if I were trading Emini futures, with slippage values of $12.50 per side, I would be somewhat concerned. As we'll see later, for that particular market, we really want average trade values up in the $200 range or so, which is much more comfortable.

This value, in particular, is a huge tip off that the system you are considering purchasing from that vendor who somehow managed to find your email and send you that juicy sales letter might not be all that it seems. I've seen quite a few commercially available systems with terribly low average trade values. If your system can't make enough to cover costs each trade, it's definitely not going to be able to make you any profits, no matter how awesome it looks in hindsight. Unfortunately, in a lot of cases, system

developers don't necessarily trade the systems they develop. Because of this lack of experience, they end up trying to push systems that just can't practically be applied, and more often than not, it shows up in the average trade value. If money could be made off of hypothetical systems, life would be a lot easier, and average trade values of $5 would mean something. Unfortunately, real life is much harder than this, so when you see low average trade values, it's a huge red flag.

Win/Loss Ratio

This is another of the more important metrics to pay attention to. The win/loss ratio is the average winning trade divided by the average losing trade. A value of 1.0 (like our example system) shows that on average winners and losers will be about the same size. So in a case like that, your system needs to have an accuracy number significantly higher than 50%, or it's just going to churn. Win/loss values larger than 1.0 mean that winning trades are larger than losing ones, and values less than 1.0 mean that losing trades are larger than winning ones. This will have a huge impact on how your system makes money over time, and how difficult periods manifest themselves for the system.

Think about a very high win/loss ratio, let's say around 3.0. That value really means that one winning trade will make up for three losses. So if you have decent accuracy, it's very easy for that system to climb out of a hole. The converse situation would be with a very small win/loss ratio, like 0.25. In that case, it means that a winning trade is only 25% the size of a losing trade. Or, to rephrase, one loser will wipe out four winners. In this situation, if you get a few losers in a row, your account is going to be underwater for a long time, as it will take a ton of winning trades to pull the account back up to positive territory.

The win/loss ratio is intimately related to the accuracy number. Systems with very high accuracy tend to have lower win/loss ratios, and systems with very low accuracy tend to have higher win/loss ratios. As we develop our systems, one of the challenges we will face is how to try and maximize both the accuracy and win/loss ratios together, which is easier said than done. As traders, we like high accuracy systems, as it's nice to hear the cash register ring. However, if the accuracy is so high that it results in a miniscule win/loss ratio, a few losers can vaporize a whole year's profits, and that is definitely not a fun situation to be in.

Going the other way, we find the classic trend following systems. These systems often have very low accuracy, like around 30%, but when they hit a winner, that winner is huge. I tend not to develop these types of systems, for the simple reason that I get stressed out when my account stays underwater for too long a period of time. Obviously, part of trading is being underwater, but some of us can take it better than others. Richard Dennis, the master Turtle himself, was one of the best at being able to deal with that situation. It was good for him that he could, as he was a big time trend follower. If memory serves, one of his funds ended up losing 80% of its value over a matter of years while his system kept trying and failing to latch onto a big trend. Eventually he did hit a few big winners and was able to pull himself out of the hole, but I suspect many of his investors weren't able to keep their cool in the same way. It takes nerves of steel to keep trading a system that loses money day after day for years, so if you set out to be a trend follower, you need to be prepared for that. If you can't take that sort of pressure, then you can't trade those huge moves. But that's OK – you can still make money trading the small stuff. The point is that these numbers will paint a picture for you about how the system will make its profits, and not only do you need to verify that your method is doable from a technical standpoint, but also from the standpoint of your own personality. Systems won't work if the trader bails out in the middle of a losing period.

If you take a quick look at the example system report, you'll see that our stock trading system has an accuracy of 68.95% with a win/loss ratio of 1.0. We also know that it has taken lots of trades over our test period (just under 900). We don't need to know anything about the rules to know that this is a relatively shorter term approach that makes money mostly by being correct about market direction. As our winners are the same size as our losers, we know that it doesn't swing for the fences, so it's not trying to ride the big moves. So if we were going to classify this system, we can deduce that it is a short-term direction picker, probably holding a position for a few days at a time and then getting out.

Largest Win

This is the dollar value of the most profitable trade taken during the test period. When you look at systems that have taken a ton of trades, this value doesn't tell you too much. However, if you are looking at patterns with only a handful of trades, this can be

an important number to watch. In that situation, you want to make sure that all your profits didn't come from one big trade.

Largest Loss

This is the dollar value of the largest losing trade taken during the backtest run. It is an important value from the perspective of risk management. At any given moment, your account can suffer a loss of this value on the very next trade. You need to make sure that you are trading with enough funds so that a loss of that magnitude isn't a big deal from the perspective of the account. When it comes to stocks, or index futures like the Emini, it also gives you a good gut-check number, as the maximum loss tends to happen during market crashes. Would you be willing to keep trading your system after absorbing a loss of that magnitude, keeping in mind that it's the largest loss *per contract* traded? In the case of an Emini daily system, you might find a value here close to -$10k. If you were trading 5 contracts, that's a $50,000 loss in one trade. If you go ahead and implement that method, you need to be prepared to absorb that and keep trading. Again, trading systems only work if the human behind them doesn't turn them off when things get ugly. More often than not, the largest winning trades follow on the heels of the largest losing trades, so turning a system off to try and protect your account from further damage tends to worsen the pain rather than relieve it. Trading isn't for wimps.

Longest Winning Streak

This number tells you how many trades in a row the best winning streak lasted. High accuracy systems tend to put together some impressive numbers in this category. Our test system was able to hit 12 in a row at one point.

Longest Losing Streak

This number is the opposite of the last metric. It tells you how long the worst losing streak lasted, measured in number of trades. Since our test system is very accurate, it has only taken 5 losers in a row. Of course, this doesn't mean that it won't someday suffer a longer losing streak than what's shown in the report.

Maximum Shares/Contracts Held

This shows the maximum position size the system put on during the test run. It won't be useful unless you are specifically testing a system that varies its position size. Most

development will take place assuming a fixed position size, in which case you'll see a value of 1 here. In our example system, I set the system tester to run on 100-lot values, so in that case this value represents one 100-lot block.

Gross Profit

This value shows how much profit the winning trades added up to over time, without considering any of the losing trades. Wave59 includes it for reference purposes, but to tell you the truth, I hardly ever look at this value or the next one.

Gross Loss

This number shows the total aggregate value of losses taken by the system over the test period.

Profit Factor

This is the gross profit number divided by the gross loss number. A trading system will run through a chart and will make profits and then take losses. It happens in a sort of two-steps-forward, one-step-back fashion. This number gives you an idea of how that process really works. A value of 2 would be more like two profit steps against one loss step, and a value of 3 would be like three profit steps against one loss step. Higher values are more desirable, as they show that it's very easy for the system to generate profits.

Annual Return

This number calculates profit as an annual return, in percentage terms. Although not shown, when running Wave59's system tester, there is a place to enter the starting account value. That account value is then used as the basis from which to compute the annual return. This number, in Wave59's system report at least, does not account for compounding profits, and assumes that all profits are removed from the account each year, except for that starting account size. We will look at how to compute compounded returns later.

Max Drawdown

This is the most important measure of risk in the entire system report. As a system is generating profits, the account value will be rising at a nice rate. Then, a losing period

will hit, which results in a negative shock to the equity curve. This negative shock is called a **drawdown**, and your system will experience many of these throughout its life. Some drawdowns are larger than others, and this metric tells you how large the worst one ever experienced was. A drawdown is the amount of money lost by the system from an equity high, all the way down to the lowest point before the equity curve recovered to make new highs. The lowest point is measured intraday, so this value includes losses that were suffered during the middle of the trade, but not necessarily fully realized in the account.

This is another important gut check number. If you started trading at the worst point in time, how large of a loss would you see the account absorb before it was able to recover and make profits? The more important question is, would you be able to handle that loss without pulling the plug on the trading system?

One of the things you'll find in trading is that Murphy's Law is more than just an idea. It's real. Every time I have started trading a mechanical system, I have suffered one of the worst drawdowns in history within the first few months of trading. This has happened with *every single system* I have ever traded live. Part of this has to do with traders being most exuberant at market tops, right before corrections. That's also the time when most new systems get launched. Remember that private fund I mentioned earlier? We started trading it in August, 2008, right before the market crash. Guess what the system did right then... Yes, bought, and held. Trial by fire, to say the least.

This is one of the places where I hope you take advantage of some of what I've been through. You need to make two assumptions with your systems. First, you must assume that the worst case maximum drawdown hasn't actually happened yet. Your system report will give you a drawdown number, so use that as a guide, and add 50% to the top to try and get a reasonable estimate for the real worst case. Second, you must assume that you will experience that worst case maximum drawdown in your own account immediately after beginning to trade the system live. Not eventually, but immediately, as in the first trade you ever take.

As you prepare to take your system live, if you are working under those risk assumptions, they will keep you from making a stupid mistake, like trying to trade an aggressive approach with an account that is too small. The simplest way to determine the correct minimum account size you need to implement a system is to go through the

following process. As a first step, sit down and ask yourself what number represents the worst case percentage loss you would be willing and able to accept in your account, without resorting to turning your system off and running to hide in the mountains. Could you accept a 20% loss and still be able to trade this system? What about 30%? What about 60%? Be honest with yourself at this stage. You're a trader, so you need to accept some risk, but you don't need to accept so much that you can't sleep at night, and you definitely don't want to develop ulcers because of your trading system. That's not going to help anyone.

This is a personal process, and everyone is going to have their own risk number that they can tolerate. For me, my number is around 40%. When I've done this exercise, I felt that I could lose 40% without getting too worked up, and manage to keep trading the account. I'm not going to want to be down that much every day, but now and then, ok, I'll accept it. You may be more conservative than me, and have a smaller number. You may be like Richard Dennis, and be twice as aggressive. Whatever your number is, make sure it's correct for you, because I guarantee you will experience that magnitude of drawdown before long. I've definitely been to -40%, and it has happened more than once. So be honest with this step, and remember that markets are more volatile now than they were in the past, which means everything happens faster and to a larger degree.

Once you've got this number, then go back and look at your worst case drawdown value. Your system report will have given you a max drawdown number, and you will have increased that value to account for the true maximum drawdown number, which hasn't happened yet. Once you have both the maximum dollar drawdown value and your worst case acceptable percentage loss, you can then compute how large an account you need in order to keep the drawdown to the percentage number that you've just calculated:

$$\text{Required Account Size} = \frac{\text{System Max Drawdown (in dollars)}}{\text{Personal Max Drawdown (in percent)}}$$

Let's do an example with our test system. Our maximum measured drawdown was $3037. Let's assume that we under-measured this and that when the next worst case happens, it will be 50% worse. That gives us a real estimated drawdown value of $3037 + (50% x $3037) = $4555. That's the Murphy's Law-adjusted drawdown value. Next, having gone through our process above, we've come up with some percentage value that represents our "uncle" point, where we don't want to be trading this system anymore. For this example we'll use my number, which is 40%.

Therefore:

$$\text{Account Size} = \frac{\$4555}{0.40} = \$11{,}387$$

This means that, in order for me to trade a 100-lot using this system, I need to have $11,387 in my trading account. If I were more conservative and had a lower pain threshold, the value would be larger. If I were Richard Dennis, and ate glass for breakfast, the value would be smaller.

What happens if you don't actually have enough money to trade the system? In that case, you have three options. Your first option is to trade the system anyway, which is what most people do. That solution will lead to you blowing your account out and getting out of the trading business permanently. Brokers try to estimate how many people this happens to, and it's always some huge percentage of retail traders. Don't let this happen to you.

A second alternative is that you can wait. No one says that you have to trade your system today. If it's a good system, it will still be good in a year or two once you are properly capitalized. Another way of accomplishing this would be to trade using a smaller position size, if the market you are trading the system against lends itself to this. For a futures system, that's not going to be a possibility, but for a stock trading system, like our example system, no one says you have to trade 100 shares at a time. Trade a 10-lot instead, and in that case you only need $1,138 in your account in order to maintain the same risk tolerances. You're not going to make a ton of money doing this, but at least you'll be trading. Just make sure that costs don't add up to too large a percentage of your trades.

Third, you can develop other systems on other markets. Focus on markets that will allow you to trade smaller size, like stocks or Forex, and work to find a system that will allow you to trade within your risk tolerances. Do not adjust your tolerances to fit the system. Make the system fit you. There are lots of ways to trade the markets, and lots of markets to trade. With a little work, you should be able to trade something that works for your situation without having to take on undue risk.

Max Drawdown Percent

This metric is another way to show the drawdown value, and it shows when the worst case percentage drawdown occurred in the account. This may be a different drawdown period from the maximum dollar value drawdown, as it is computed using the actual account value running forward through time as profits and losses are being added to it.

Max Drawdown Date

This is the date when the maximum dollar drawdown occurred. It's useful to know when the worst case scenarios happen, so you can go back on your charts and examine them.

Max Drawdown Percent Date

This is the date when the maximum percentage drawdown occurred. Again, it can be a different date than the worst case dollar drawdown. This actually happened in our example system. Our worst percentage drawdown happened early in testing around 2001. That's because the account was smaller then, so the drawdown amount that happened ended up being a larger percentage of the account than the ones that happened later, after the account had grown a bit. You'll see that the maximum dollar drawdown happened in 2008.

Again, when judging these values, remember to apply Murphy's Law. It's best to use the maximum drawdown dollar value, and assume that it will happen on the very first trade, before any profits are built up to create any sort of padding.

Longest Time Underwater

Maximum drawdown values tell you how much money your account will lose in the worst case. Longest Time Underwater tells you the amount of time that you will spend underwater during those bad periods. It is measured from the bar the account first went into drawdown, and stops on the bar when the account broke back to new equity highs

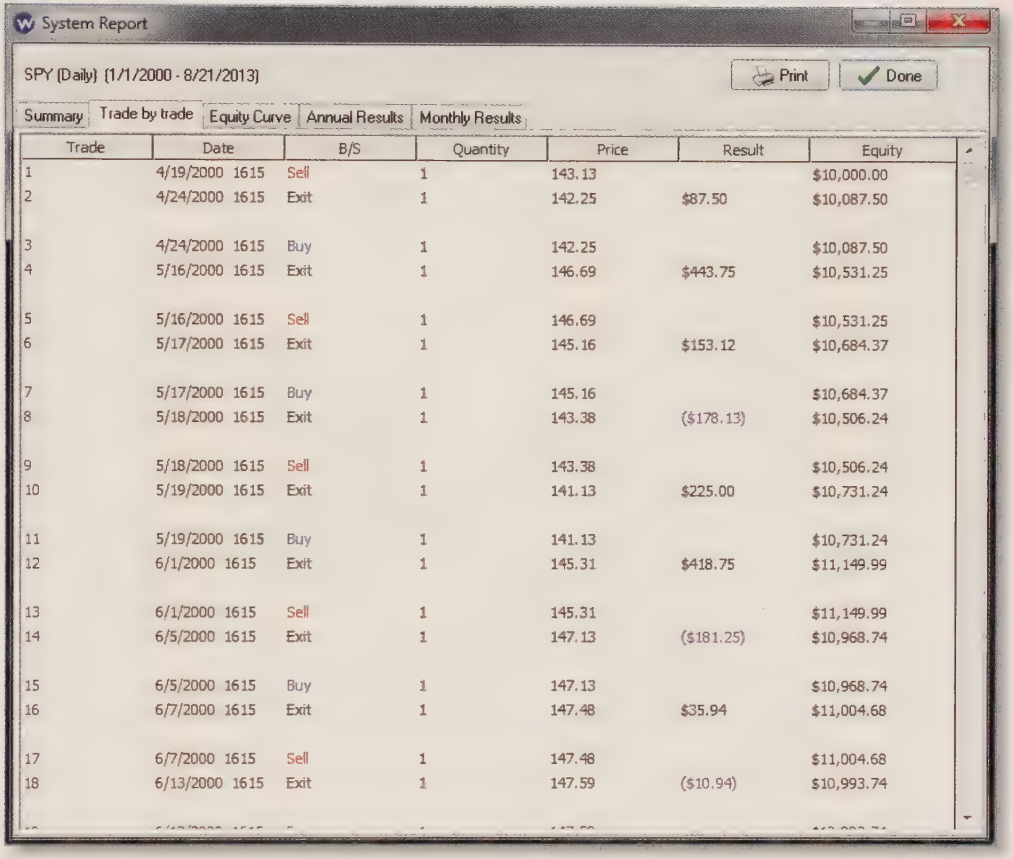
past that initial point. It is measured in bars, and so it will reflect whatever time frame you are trading. In our example system, we had to sit through a 112 bar period where we weren't breaking to new equity highs. As this is a daily chart, showing only trading days, that's about half a year. You'll need to be prepared to keep trading despite being underwater during these periods, so definitely check these numbers to get a feel for what could happen.

Longest Time Underwater Date

This is the date when the account finally broke out of its longest underwater period. It's there so you can go back and find that place on the chart as you are working through your system development.

So those are the metrics in the *Summary* tab. After you work with them awhile, they'll become second nature to you. Some of them are more important than others. The primary ones that you'll always want to keep on top of are: Profit, Accuracy, Win/Loss Ratio, Average Trade, and Drawdown. These five metrics tell you the primary story behind how your system performed.

Trade by Trade



Trade	Date	B/S	Quantity	Price	Result	Equity
1	4/19/2000 1615	Sell	1	143.13		$10,000.00
2	4/24/2000 1615	Exit	1	142.25	$87.50	$10,087.50
3	4/24/2000 1615	Buy	1	142.25		$10,087.50
4	5/16/2000 1615	Exit	1	146.69	$443.75	$10,531.25
5	5/16/2000 1615	Sell	1	146.69		$10,531.25
6	5/17/2000 1615	Exit	1	145.16	$153.12	$10,684.37
7	5/17/2000 1615	Buy	1	145.16		$10,684.37
8	5/18/2000 1615	Exit	1	143.38	($178.13)	$10,506.24
9	5/18/2000 1615	Sell	1	143.38		$10,506.24
10	5/19/2000 1615	Exit	1	141.13	$225.00	$10,731.24
11	5/19/2000 1615	Buy	1	141.13		$10,731.24
12	6/1/2000 1615	Exit	1	145.31	$418.75	$11,149.99
13	6/1/2000 1615	Sell	1	145.31		$11,149.99
14	6/5/2000 1615	Exit	1	147.13	($181.25)	$10,968.74
15	6/5/2000 1615	Buy	1	147.13		$10,968.74
16	6/7/2000 1615	Exit	1	147.48	$35.94	$11,004.68
17	6/7/2000 1615	Sell	1	147.48		$11,004.68
18	6/13/2000 1615	Exit	1	147.59	($10.94)	$10,993.74

Figure 2.2: Example System Report - Trade by Trade

This tab displays a listing of every single trade taken by the system during the backtest period. It's available to you in case you want to dig into the details and do nitty-gritty analysis. It's also extremely useful if your system isn't performing as you expect, and comes in handy when debugging any programming errors that may currently exist within your system code. In Wave59, you also have the ability to export the trade details to a file by right clicking and selecting the "Export" option. This will allow you to export your results to Excel for further analysis.

Equity Curve

The next tab over is the *Equity Curve* tab, which shows a graphical plot of the equity curve over time.

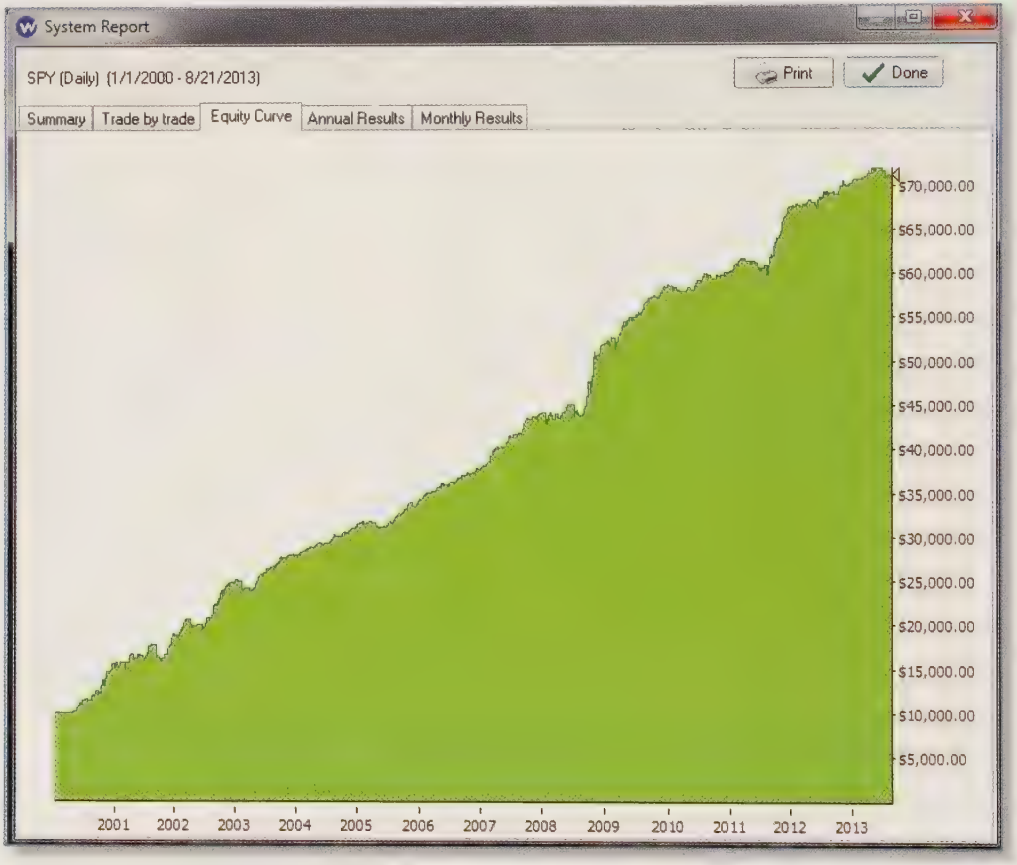


Figure 2.3: Example System Report - Equity Curve

This plot starts at the beginning of the period, and shows what the account looked like as it moved forward, including all open trades. Notice that it doesn't start at zero. Rather, it starts at whatever the initial account value was set to when the system tester began its simulation. I frequently like to look at just the profits themselves though, and to do that, you just launch the system tester with a tiny initial account value, like $1.

Notice the change:

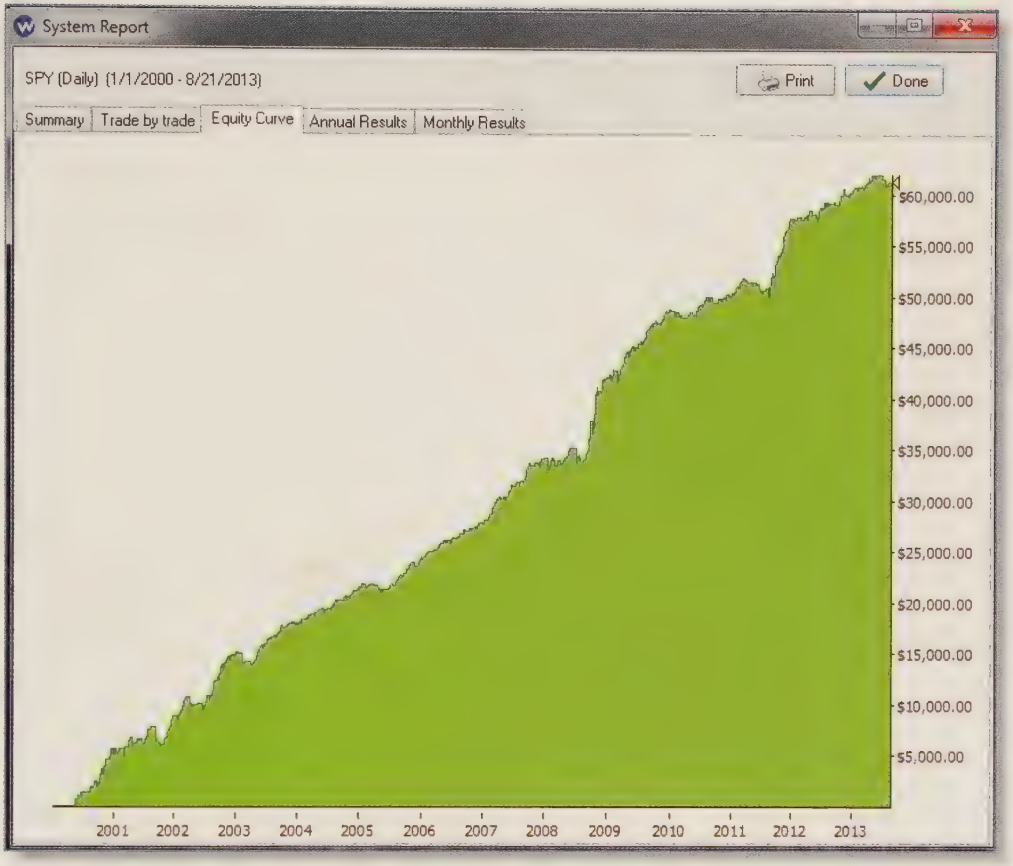


Figure 2.4: Example System Report - Equity Curve without Initial Equity

Note that when running a system report using a small initial account value, some of the report metrics, such as annual return and drawdown percentage, will get blown out of proportion. You'll probably see that as we begin working with different systems in later chapters.

As I've developed systems throughout the years, this one plot has become probably the most important tool I use to evaluate a system. It shows you the character of the system, and gives you a much better feel for how the system performed than trying to interpret all those metrics. You still need the metrics, so don't throw them out (especially if you just learned them!), but this is where I go first when I'm trying to understand a system.

In order to evaluate an equity curve plot, you need to understand what the ideal equity curve would look like, which is a completely flat, upward-sloping line:

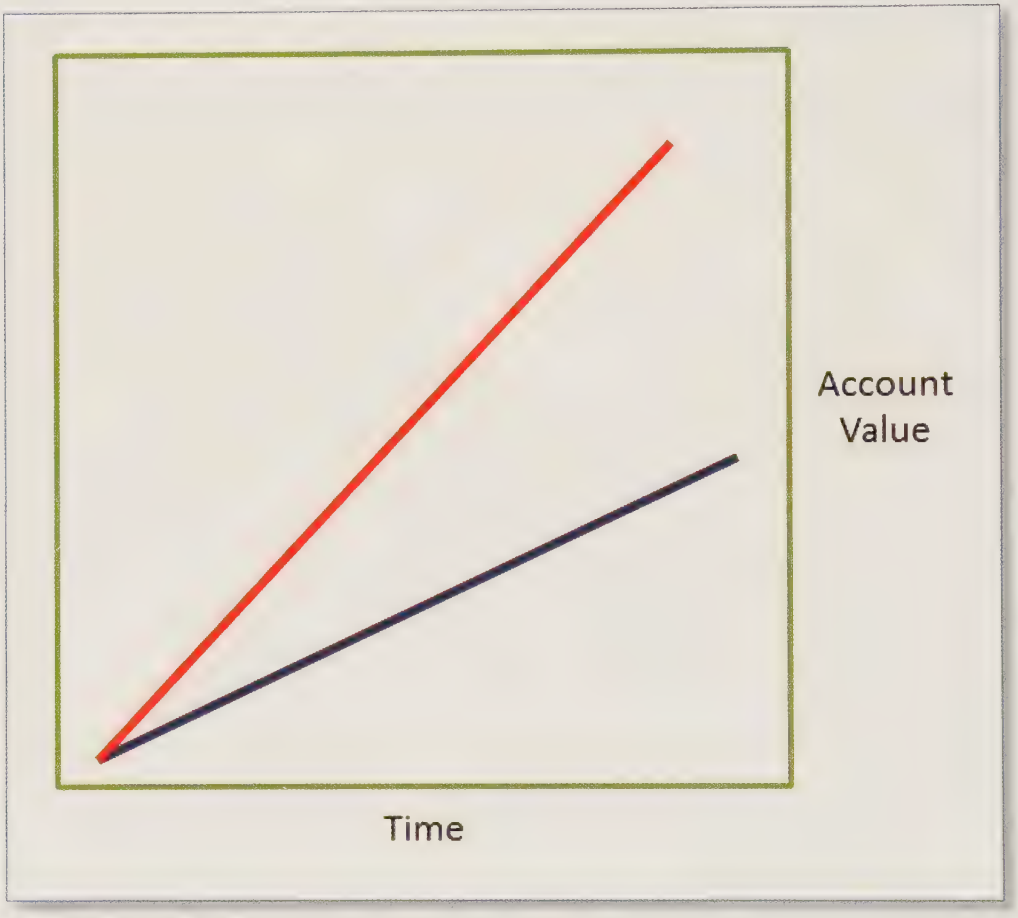


Figure 2.5: Ideal Equity Curve

The graphic above shows two ideal equity curves. That might sound confusing at first, as you'd think that there could only be one ideal, but that's not really true. What makes these two lines ideal is that neither one of them have any variability, which means that there is no uncertainty anywhere to be found. Each day you trade one of those systems, you are guaranteed to make a profit, and you know exactly how much profit you will make. That's why they are straight lines, rather than the wavy, stair-step equity curve on the previous page. The only difference between these two curves is that the red line makes more money per trade than the blue one.

Obviously, there aren't a lot of systems that have straight-line equity curves like this, at least in the real world of trading systems. But this is what the holy grail of trading systems would look like if you could find it.

So when we evaluate our system results, one of the things we want to see is an equity curve that mimics these ideal lines as much as possible. The example system we've been discussing in this chapter is a good example of that. Although it does have its bumps, for the most part it's a nice, smooth line that gains ground in most periods. It's not perfect, and deviates from the straight-line ideal here and there, but the key thing is that those deviations are relatively small. As a side note, this is an actual system that I'm trading right now, so although I'm not planning to reveal it's logic (at least not in this book), this isn't a hypothetical example.

Let's take a look at some other equity curves, and see what they have to tell us.

Figure 2.6: Bad Equity Curve = BAD system

Figure 2.6 is an example of a system that you really wouldn't want to trade. It made a tiny bit of money in the first few weeks of trading, then got completely hammered through 2003, at which point it started going sideways-to-down. We really don't need to examine the summary results in much detail with a system like this, because the equity curve tells us all we need to know. Wave59 plots negative equity in red when drawing these graphs. That's the account bleeding out.

Here's another:

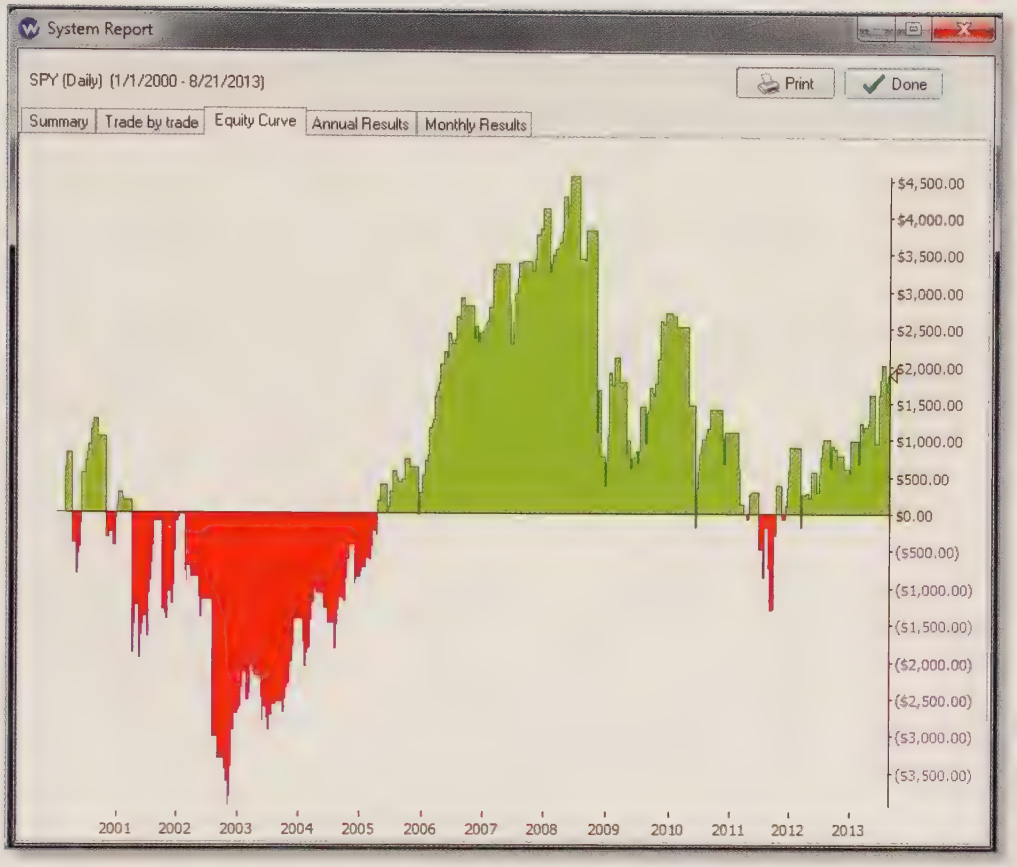


Figure 2.7: An Equity Curve of a System with No True Edge

In this case, we're doing better than before, but you can see that there's no upward slope to the equity curve. It really just oscillates above and below the zero line. A system like this basically has no edge, and is just churning back and forth. It's not necessarily going to tank the account right away, and if you had started trading this one in 2003, you'd have had a really good run up until 2008, but over time, this sort of system is not going to work.

As an aside, this particular equity curve also demonstrates how important it is to test your systems on as much data as you possibly can, as well as the importance of having as many trades as possible covered by your reports. There are parts of this equity curve that make the system look like a winner, such as the 2003-2008 period. There are other parts that make it look like a terrible loser, such as 2000-2003, or 2008-2011. If you are

only running your reports out a few years, who knows which particular period of time you might land on. If you're lucky, you'll see the downward periods, but odds are that you'll actually end up testing it up through 2008, thinking that 5 years of data is plenty. In that case, you'd have ended up trading this system live all the way down through a terrible losing period, you would still be underwater in 2013, and the system might never break to new equity highs. So keep that in mind, and use as much data as you can when running these reports.

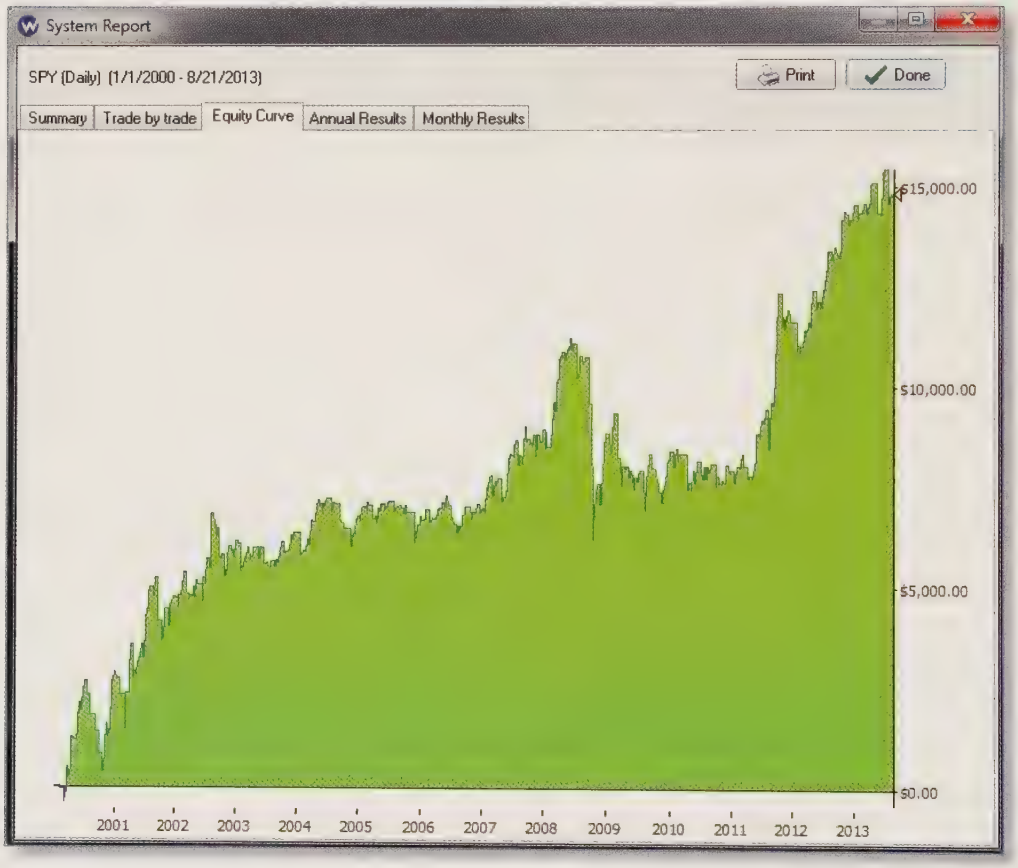


Figure 2.8: Equity Curve that Shows Some Promise

This equity curve is another example of what you'll find as you start developing systems. This one tells you that you might be on to something, but that you need to put a little more work into the method. First of all, the curve is upward sloping, which shows there's some edge to the approach. That's what you want to see when you program

your ideas. It's not a straight-line, by any means, but there's some traction here despite some difficult times. In particular, this system is having trouble with the period 2004-2007, as well as 2008-2011. That gives you some dates to go look at and see if you can come up with some ideas why your method just isn't being able to perform profitably during those times. A system like this can sometimes turn into a system like the first one we looked at, with an awesome equity curve, so this is the market giving you a thumbs-up on the general idea, just maybe not the implementation.

Hopefully that gives you an idea of how to use the equity curves to gain an intuitive sense of how healthy your approach is in general terms. Humans are good with pictures, and equity curves are system performance results in picture form, so it's easy to latch onto them. We'll end up talking about equity curves with regard to all the systems we look at in this manual, so you'll get lots of experience working with them by the time you're through.

Annual Results

Moving along, the next tab in the system report is the *Annual Results* page. This shows you the breakdown of your profits year-by-year.

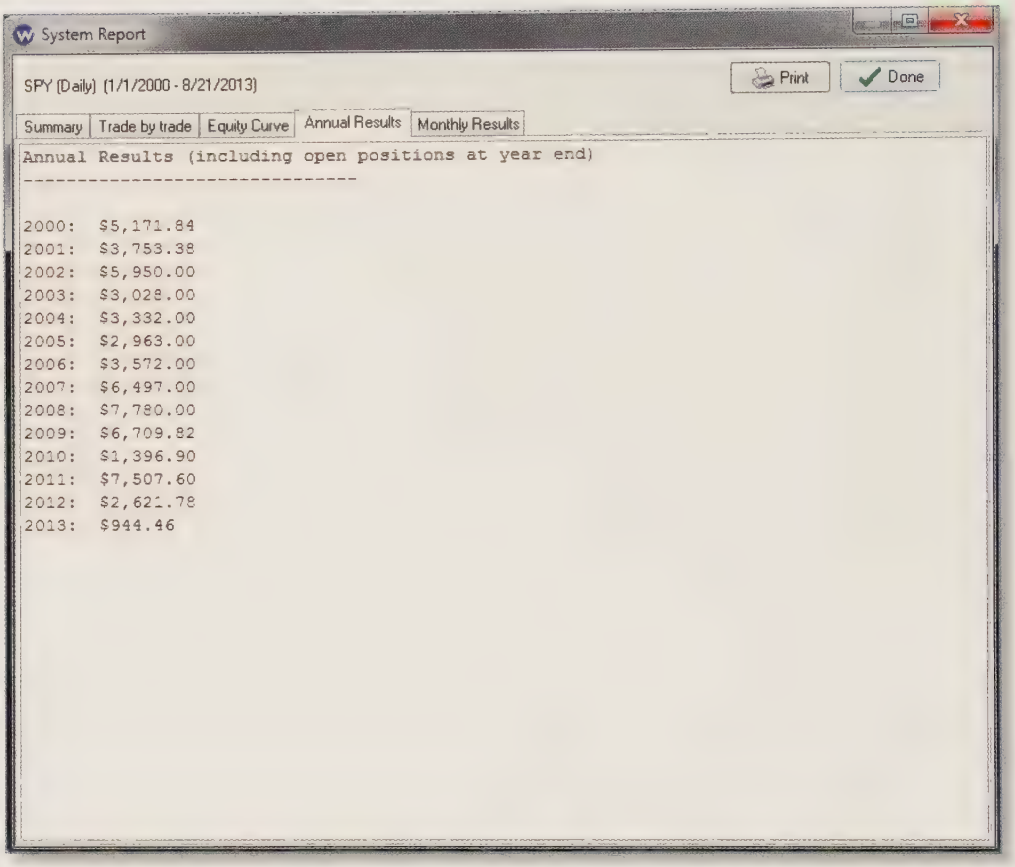


Figure 2.9: Example System Report - Annual Results

You can see what it looks like for our test system. Ideally, you want to see profits in each year, and they should be around the same size. That's not always possible, and there are some years that end up being great years, and others that end up being poor years. In our case, our system handled 2008 and 2009 wonderfully, but didn't do nearly as well in 2010. But all years were profitable, and that's a great situation to be in.

Monthly Results

The next tab over, *Monthly Results*, is the exact same sort of thing, but breaks things down on a monthly basis.

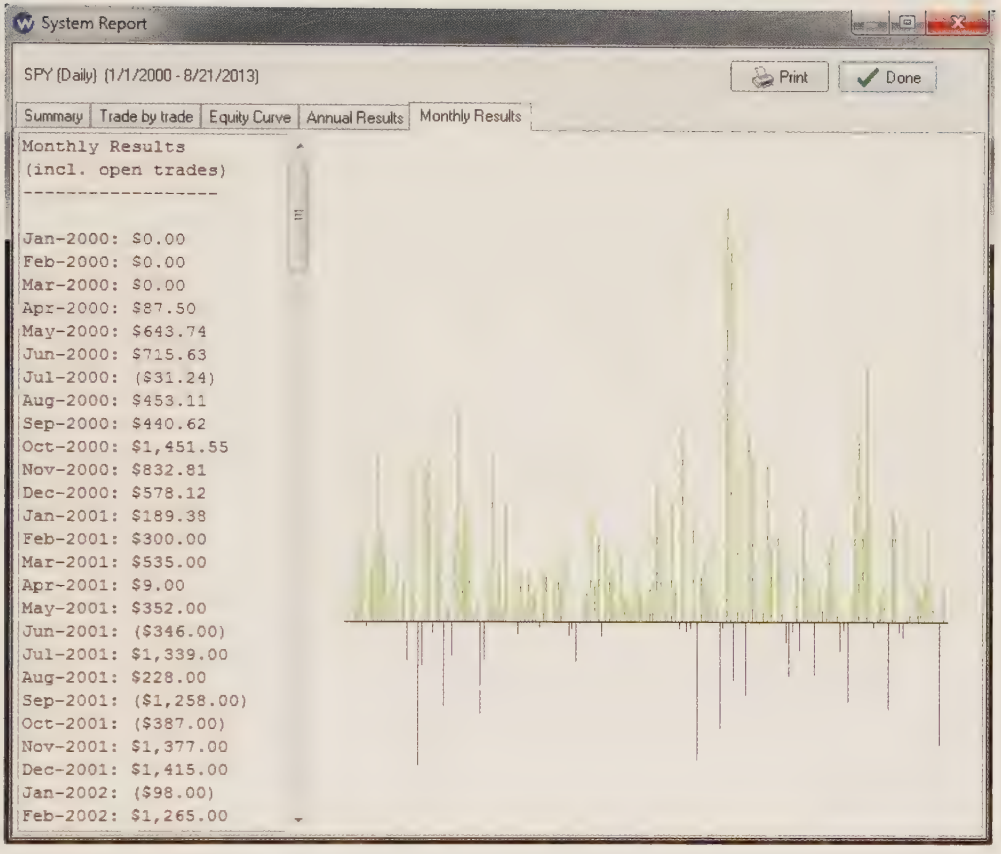


Figure 2.10: Example System Report - Monthly Results

The data is presented both in tabular form on the left, as with the annual results page, as well as in histogram form. In the histogram, green bars are profitable months, and pink ones are losing months. Market volatility will have something to do with how this looks, and in our case we have two huge green spikes in September and October, 2008, which was during a crazy period for the stock market.

There are plenty of other ways to dissect a system, and future versions of Wave59 will add all sorts of interesting modules to this basic format. But the real meat of system development comes down to the tools we've discussed in this chapter. Between the five key metrics (total profit, accuracy, win/loss, average trade, and drawdown) and the equity curve plot, you have everything you need to properly evaluate mechanical systems. There's obviously more to system development than just this, and as we

progress we'll discuss quite a few more topics, but you can consider these items as the gatekeepers that must be satisfied for a system to gain access to your live account.

For lack of a better word, there's just a lot of crap floating around in this industry. One out of every hundred ideas might actually be worth something, and those nuggets usually come out of your own sweat equity, rather than from the classified ads in the back of magazines. If you put your hard earned money into a bad trading system, that system will magically move the decimal point on your account to the left, transforming a large amount of money into just a tiny pile in a short amount of time. Conversely, if you put your money into a good system, it will turn it into a larger amount, and over time, the profits may be enough to retire and live well off of. Being able to detect whether or not a particular system is a good one or a bad one is a critical skill that you'll need to develop as you work with systems, and these tools form the basis for that process. As we work with some of the trading systems in this book, you'll learn more about how to do this.

3. William Tell Bonds

The William Tell Bonds system was the very first system I traded successfully, and was the product of some very intense development work back at the very start of my trading career. William Tell is a folk hero of Switzerland, famous for shooting an apple off his son's head with an arrow. Something like that can only be accomplished if one possesses exceptional accuracy, and as you'll see, this system is all about accuracy. Before we get into some of the details of the method, let me provide a little background to explain my mindset and goals at the time.

I had gotten interested in trading back in high school, when a friend of the family let me ride along in a big options trade he had put on by giving my one of his options contracts. This guy was an ex-money manager from New York, who got completely fed up with the rat race and ended up moving out to the sticks in New Mexico to live a different sort of lifestyle. He was a really nice guy, but a little extreme, and his idea of a different lifestyle was to go be an independent lumberjack, cutting down trees and selling firewood to the locals. He drove a beat up old pickup truck, and had a habit of "liberating" abused dogs from a nearby Indian reservation, which meant that there was a pack of a dozen dogs that would follow him wherever he went. Anyway, selling firewood for a living doesn't exactly put a lot of money on the table, and to supplement his income he'd put four or five big options trades on each year. During one of these trades, he gave me one of his options contracts, and we followed the whole thing along together. We didn't make money with this specific trade, but I was completely hooked.

Throughout high school and college, I read and studied everything I could get about trading. I used to save all of the direct mail pieces that the system vendors would send out, and would spend my time studying their charts and scheming about how much money I was going to make once I could start trading for myself. Once I graduated college, the first thing I did was manage to borrow $25,000 in cash off of credit cards, and opened up a futures trading account. The next thing I did was purchase charting software, a data feed, and a $3500 mechanical trading system from one of the vendors I liked. This system was based on the Taylor book method, and unlike the systems in a lot of the other sales letters, the advertised results from this one were based on real money and actual trades. It had an awesome equity curve, and it had done really well on the S&P 500 futures, which was my favorite market at the time.

Unlike today, there weren't electronic markets, and everything was done in the pit. The S&P pit was the biggest, most active, and wildest market. It traded at $500 per point, and moved enough to make thousands of dollars per day if you could get the direction right. The money was there for the taking for those who knew how to do it, and I had an awesome tool to do just that. I was psyched that first day, and triple checked everything before I called my orders in, making sure to follow the system exactly. As long as I did what it said, and didn't make any beginner mistakes, I'd be good. Aside from something like that, what else could go wrong?

To answer that question, plenty. Volatility is great if you're right with your trades, but it can cut both ways, and the S&P 500 futures might not have been the best choice of markets on which to start my trading career. My system got *crushed*. In a matter of two weeks, I had lost more than half my account. It wasn't a slow grind either - it felt more like the bottom just fell out of the account all at once. I had lost enough money that I couldn't trade the S&Ps anymore, as I couldn't post the margin to trade even a single contract.

As it turns out, the "real money" equity curve that my expensive trading system had sold me on turned out to be not so real. In fact, a few years later the CFTC ended up prosecuting both the developer and the advertising company responsible for cooking up those juicy sales letters. The equity curve was fabricated, based on hypothetical testing in the past. No one had actually placed any of those trades, and despite claiming that prospective customers could audit those brokerage statements, there really weren't any brokerage statements to audit. The two guys basically got a slap on the wrist, having to pay a fine that was only a fraction of the proceeds they made off of their scam, and none of us system purchasers received any kind of compensation for the fees we had paid to purchase the system, let alone for any losses suffered as a result of trading it.

So I learned a few very important lessons in the first two weeks of trading. The first, and probably the most important, lesson was to never trust anything you haven't verified yourself. Don't take anyone's word for anything in this industry. It's great to learn, and buy books and systems, but before you put money on anything, make sure it really can do what you think it can do. Would it have killed me to paper trade this method for a couple months for verification purposes? Second, don't trade with an undercapitalized account. If you do, it won't take long for regular price action to knock you out of the market, and possibly out of trading in general. I should never have been in something as

crazy as the S&P with an account the size I was running. Third, it's probably not a good idea to buy black-box (where the logic is not disclosed) trading systems. And finally, it's probably not a good idea to even read those black-box trading system sales letters, as they're not exactly based on facts.

This brings us to my mindset after two weeks of trading, back when I was fresh out of college. I had around $12k in the account left, and a huge credit card debt hanging over me. I knew that if I ended up going out and getting a job, that would probably be the end of my trading career, at least for a decade or two. So if I wanted this to work, I really had to find a way to be able to trade right then, and earn back all that money. I sure as heck wasn't going to be using that Taylor trading system though, and I wasn't ever going to buy another one of those black-box systems after my experience with this one. My only option was to develop my own approach, and for the next 3-6 months, I lived, breathed, and ate trading systems.

At the time, being a little shell shocked and mostly concerned about my survival in the markets, I had one overriding focus as a developer. I wanted my system to be accurate above all else. I didn't want to take any losses, and I needed a system that I could trust to make enough each month for me to stay afloat, given the fact that I now had bills to pay and no income to speak of.

My solution ended up being the William Tell Bond system, which was based on an exit strategy that I had learned in one of Larry William's courses. In a video course of his, he ran through a dozen or so S&P patterns, and always exited the trade at the first profitable open. So after you got into a trade, you'd just watch every opening price each day afterwards. As soon as an open came in even one tick in your favor, you closed the trade. Along with that, you would apply a very large stop loss. The result was an exit strategy that made it very easy to exit with a profit, and made it very difficult to exit with a loss. Of course, when you did take a loss, it was huge, but let's not worry about the implications of that just yet.

An important thing to realize about system development in general is that the way you exit your trades has more to do with the performance of your system than the way you enter them. In fact, and I have proved this with my own research, if you can develop a good enough exit strategy, you should be able to make money trading it even using random entries. Entries are an order of magnitude less important than exits. Of course,

in this industry, the majority of the focus is on entry signals, with exits being mostly an afterthought. This helps to explain why most people can't create systems that actually work. They have been brainwashed into thinking that the way to succeed in the trading game is to have a better entry point than everyone else.

Given the fact that I wasn't able to actually replicate any of Larry's system results with the patterns he developed, I figured that I probably shouldn't push my luck and should accept his exit technique and develop the rest myself. I targeted the 30-year Treasury Bond futures, which were my new favorite market. They moved quite well back then, and the margin requirement was small enough to allow me to trade them comfortably, given my tiny account size.

In this chapter, we're going to look at some of the work that I had done with the William Tell system. I have fond memories of this approach, because this was the system that saved my ass, pardon my French. I started trading it in 1997, and continued to trade it through my time in Chicago at the Board of Trade, all the way up to the point when I moved back to Colorado to begin work on Wave59. It earned back my losses, kept the bills paid, and held me up until I really learned how to trade on my own.

Having said that, for reasons that we'll get into shortly, I don't consider it to be a terribly robust system, and if any one of the methods we'll be looking at is going to end up blowing up in your face, this is probably the one. The system itself is based on the following key ideas:

1) The first-profitable open exit (for accuracy)
2) Bar-to-bar price relationships to determine direction
3) Day-of-the-week seasonal patterns as filters

As I was an accuracy hound, all I cared about were very high accuracy (80%+) patterns. Finding patterns like that isn't nearly as easy as it sounds, and you have to accept some difficulties in other areas of a system report if you are trying to maximize accuracy. The following is one of the patterns that went into the approach:

Pattern #14: (In English)

If Monday was a pivot low, with a lower low than both Friday's low and Tuesday's low, then buy on Wednesday at Tuesday's high plus 75% of Tuesday's range on a stop. Exit at the first profitable open, or by using a stop of 2.75 points.

Sometimes it helps to see these patterns visually:

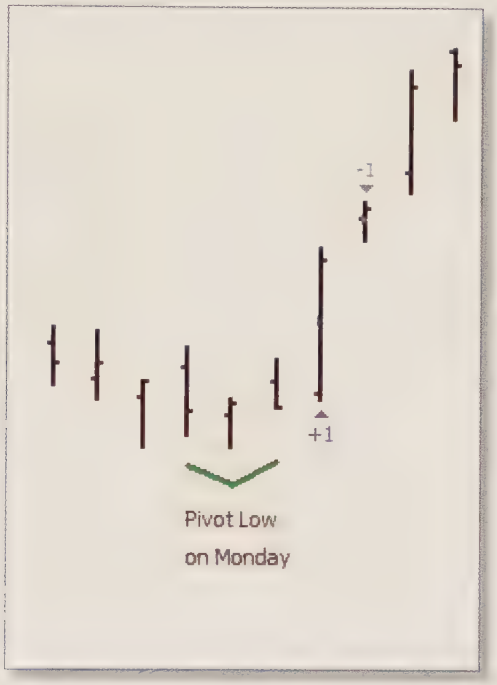


Figure 3.1: William Tell Bonds Pattern #14

The idea behind this one is to catch a weekly reversal bar, where a pivot low is made on Monday and the entire week ends up being a runner. In the picture above, we're buying on Wednesday in the middle of a break from just such a pattern. We exit at the very first profitable open, which happened one bar later at the gray arrow.

Here's the system report, run forward from January 1986, which is as far back as Wave59's integrated data feed will let me go:

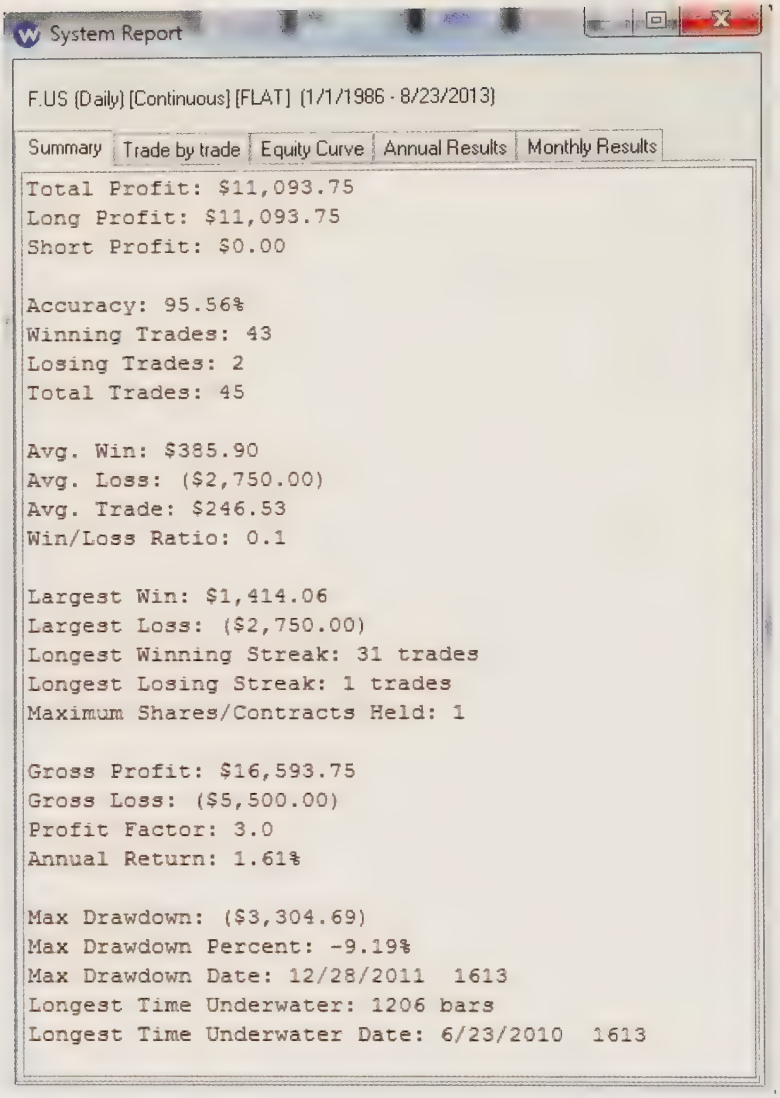


Figure 3.2: William Tell Bonds Pattern #14 System Report

Notice the accuracy number: 95.56%. Good job, Pattern 14! That's what it was meant to do. But notice a few other things, which tend to go along with this approach. First of all, there were only 45 trades in 27 years. That's not even two trades per year. 95% accuracy is awesome, but if you only get to trade once a year, that's not going to be very useful. This was the reason why I had to develop so many patterns. If you have around

20 of these all running together, you'll get enough action in the account to be an active trader.

A bigger problem with such a small number of trades is the issue of not having enough samples in your test to consider the result to be statistically significant. We touched on this in the last chapter. If you have a 100% accurate pattern that has traded only two times ever, can you really claim that there is any kind of edge there? Not really. It doesn't take much for a coin flip to land on heads twice in a row, and any sort of random trade can be right two times in a row as well. That's not going to be the case with a trade that happens a few hundred times though. So what you really want to see is a high accuracy pattern with lots of trades behind it, which, unfortunately for the developer, is exponentially more difficult to find than a high accuracy pattern with only a few trades. So when working with this sort of thing, you need to establish a guideline as to how many trades are enough, and how many aren't. Mine was 30 trades, which is about the minimum you need in order to establish statistical significance. During my development of this method, I found quite a few patterns that didn't qualify, and ended up getting thrown out. I hated throwing away 95%+ accurate price patterns, but if you can't trust the number due to the pattern only having a few instances, then it's all just an illusion anyway, and you don't want to be trading it.

As an interesting aside, one of my friends in the Wave59 community just recommended a book to me, which I finished reading a few weeks ago. It was a book on a very esoteric approach to market forecasting based on astrological patterns and geometry. The author had basically claimed to have solved the markets with this approach, and used the technique to compute many important turning points in the history of the Dow, including the crashes of 1929 and 1987, among other things. He had also used it in real time to predict a major low point, which he was very proud of, as it happened during a very bearish period of time in the media.

The method looked very interesting, but it had one small problem – it didn't happen very often. There are only about 100 years of data available on the stock market, and if you are looking at slow moving planetary cycles, you can count the number of signals you get on only one or two hands. If all of those signals match important points in history, maybe the pattern has some power, but on the other hand, maybe you are just getting lucky. Traders might be interested in a pattern like that, but if you tried publishing it in a scientific journal, the statistical significance of such a small sample size

would immediately be called into question. Compare that to a method that contains a few hundred signals that solve for all the important tops or bottoms in the history of the stock market, and you might end up with a Nobel Prize in Economics. So there's a huge difference between a method that only triggers a handful of times, and one that triggers hundreds or thousands of times.

The market is a numbers game, and it's important to remember that. The purpose of building a mechanical system and running a backtest is to determine what edge, if any, your approach provides. But, if your method doesn't generate enough trades, then all the numbers calculated by your evaluation tools are not much more than guesses. And if you're guessing, what are you doing trying to build a system anyway? More to the point, why are you trading markets? If you're going to guess, it's cheaper just to go to Las Vegas. You'll lose there too, but at least you'll have fun doing it.

Let's come back to our system report. Most of the other statistics are pretty good. $246 average trade means we can take some slippage on this system and still come out on top. A tick in the TBonds was 1/32nd (or $31.25) back when I was developing this, so minimum trade values need to be large enough to account for losing 2-3 ticks each trade. The drawdown value is nice and small at -$3,304 as well. The only issue that I see is the win/loss ratio, which is at 0.1 (actually 0.14).

Remember that the win/loss ratio is the average win divided by the average loss. In our case, it means that losses are around 7 times larger than wins. This is directly related to our exit strategy. Since we are getting out at the first profitable exit, it's very easy to take a profit, and you can see that with our amazing accuracy numbers. However, when we do take a loss, we take one at -2.75 points, which at $1000 per point is -$2750. So although losses are rare, and we've only taken two in 27 years, when they happen, they hurt.

If we reduce the size of our stop-loss, we make the losses smaller, but we also end up taking more of them. More losses means lower accuracy. So you can see how the accuracy number and the win/loss number are related. When one goes up, the other goes down. It's up to you when you are developing the system to determine what the appropriate value needs to be. In my case, I was willing to risk the couple losses in order to mostly have wins, and so I tuned this system towards accuracy.

There's an important formula that I've mentioned in other manuals that relates these two metrics in terms of what it takes to break even. Specifically:

$$Required\ Accuracy\ to\ Break\ Even = \frac{Average\ Loss}{Average\ Loss + Average\ Win}$$

If you know your average loss and your average win values, you can then figure out how often you need to be correct in order to break even in your trading. This simple formula can often be used to explain why an account is not growing, despite most of the trades being winners.

For example, there are many day traders that think they can be professional traders by scalping the Emini, trying to take half a point out each trade. The Emini moves so much that it's very easy to get your profit targets filled if you only try to get half a point. When I talk to traders like that, and I've met quite a few, I usually ask them how much they are risking on the trade, which tends to be around a couple of points. I then plug those numbers into our equation and end up with the following:

$$Required\ Accuracy = \frac{2}{0.5 + 2} = 80\%$$

For this scalper, they need to be correct on 80% of their trades, just to break even. If they want to actually make a profit, they need to be even more accurate. This is why accuracy numbers alone don't tell the whole picture. Most beginning traders would be thrilled to be 80% accurate, but being 80% accurate doesn't necessarily mean you are profitable, and being profitable is really what counts.

So if we come back to this awesome 95%+ accurate system, let's stop and see what our breakeven values are, given a 0.14 win/loss ratio:

$$Required\ Accuracy = \frac{2750}{2750 + 386} = 87.6\%$$

So in our case, we need to be 87% accurate to break even. We are beating that, but it shows you that the margin of profitability isn't quite as large as it would initially appear.

Let's take a look at the equity curve for Pattern 14:

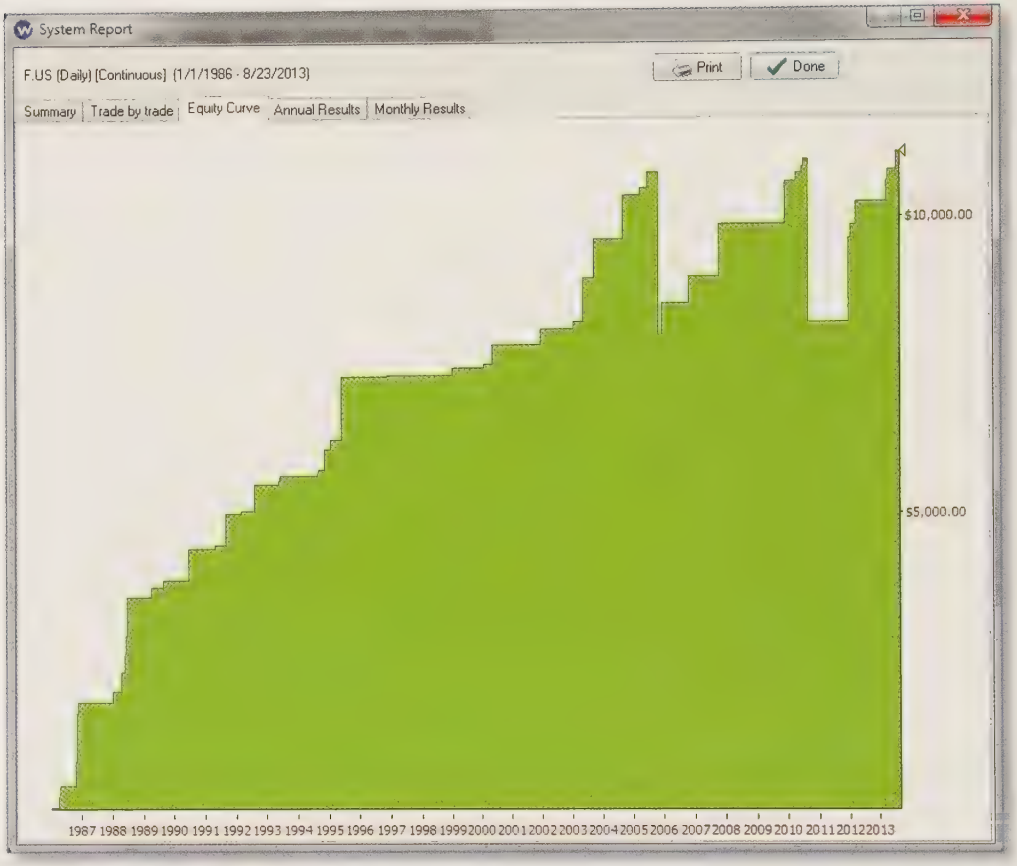


Figure 3.3: William Tell Bonds Pattern #14 Equity Curve

Not too bad. Nice, upward slope for the most part, although it looks like there were some trouble spots towards the right side. Those two dips down were the two losses that this approach took. That's right, every trade in almost thirty years was a winner except for those two, and you can see how big of an impact they had on the equity curve. This is the drawback to a small profit/large loss approach.

You can really see it in the monthly results tab:

Figure 3.4: William Tell Bonds Pattern #14 Monthly Results

Those two nasty pink lines on the histogram correspond to the losses. (I added the "Ouch!" on the chart myself – Wave59 doesn't do that automatically).

You can see the sort of system that this approach creates. When it works, it's awesome, and in fact William Tell ended up going on a winning streak that resulted in 113 winners in a row at one point. That's Gann-style accuracy, and his picture hangs in some people's trading shrines because of that level of performance. This is how you'd be able to replicate it, if you are an accuracy junkie. But the downside is harsh, as one losing trade can wipe out many winners.

You can really see the effect the large losses have when you examine patterns that just don't trade very often. Pattern #2 is a good example:

Pattern #2:
If today is an inside day (lower high and higher low than yesterday) with a close above the open, and if tomorrow's open gaps up above today's high, then we're going to buy tomorrow using a limit order set at the high of today minus 15% of today's range. We'll only take this pattern at the end of the week, on Thursday or Friday.

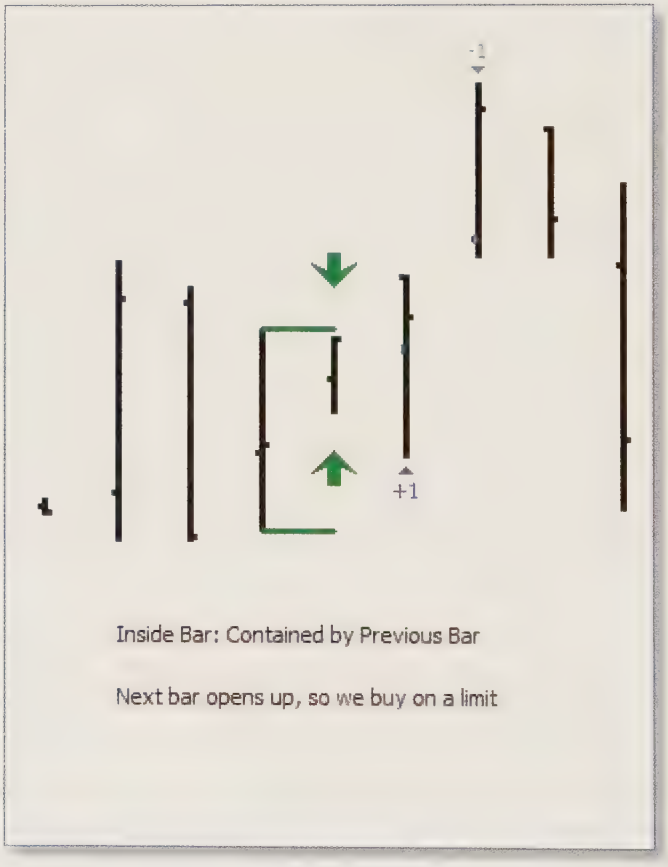


Figure 3.5: William Tell Bonds Pattern #2

There are two ideas behind this pattern. The first is based on the observation that inside bars act as coiled springs, and markets like to make strong short term moves off of those

patterns. The next idea is based on gap theory, which suggests that markets don't like to leave gaps open. So if the market jumps up with an opening gap in the direction of the coiled spring, we may sometimes get a chance to get into that move with a better price than the open, as the market will try to close the gap before continuing forward with its surge.

Here's the summary report:

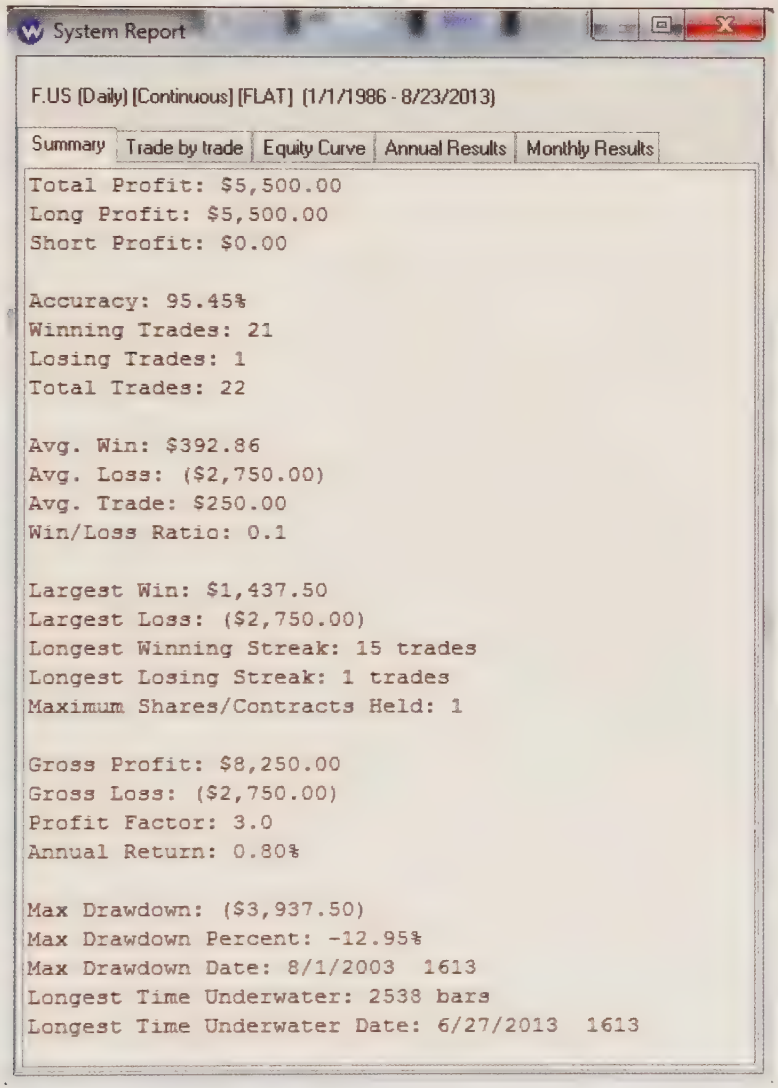


Figure 3.6: William Tell Bonds Pattern #2 System Report

Here's another 95% accurate pattern. Notice that there were only 22 trades taken. If we were developing this system today, it would not have made our minimum trade selection cut, and we would be forced to toss this pattern out. However, back when I developed this system, I used data all the way back into 1980, and there were enough inside bars back then to qualify the pattern.

This should all look very similar to Pattern #14. Let's take a look at the equity curve next.

Figure 3.7: William Tell Bonds Pattern #2 Equity Curve

This equity curve illustrates these ultra-high accuracy systems very well. That drawdown right around 2003 that I've marked is due to a single losing trade. All the other trades were winners. You can imagine how this would change if we had a couple of losses in a row, rather than just one. We'll see some examples of that shortly.

This brings up an important point with this type of pattern trading. Obviously, there's an attraction to very high accuracy systems. Yes, a losing trade can hurt, but if it only happens once every 30 years, that means that for the rest of the 29 years, the account was up. So trading a system like this is a calculated risk. If you can avoid the losses, and the odds are definitely in your favor in the short term, there is absolutely some money to be made.

As an analogy, imagine walking into Las Vegas and sitting down at a roulette table with $500, knowing that you have a 95% chance of making $25 on the next spin. That means that 19 out of 20 times, your numbers will hit, and you'll make $25. But at the same time, 1 time out of 20, you'll lose the entire $500. Would you play that game? Many people would take that bet, and would stay for a few spins. Make enough for dinner and a show, then get the heck out of there! Usually it will work out, but every now and then you'll get caught. That's a lot like how it is to trade these systems.

If you are trading for the long haul, you know that you're going to take those losses, and in that case, you have to be able to recover quickly. This particular pattern took its loss around 2003, and only managed to break to new equity highs in 2013, so you don't want to have to be down for ten years making up for one trade. The way around that is to trade a bunch of these patterns all at once, and hope that the losses from each pattern don't cluster. That way you'll take a loss every now and then from one pattern, but all the other patterns will be generating profits during that period to keep the boat afloat. Safety in numbers, so to speak. We'll look more at that in a minute.

First, let's take a look at one of the better patterns in the collection.

> *Pattern #12:*
> Compute a 2-bar simple moving average of the highs, and another of the lows. If a bar closes up (above the previous close) add the difference from today's high to today's close to the average of the highs. If a bar closes down compared to the previous close, then subtract the difference from that bar's close to the low from the average of the lows. This creates trading bands. If tomorrow opens below the lower band, then we're going to buy on a stop at the level of that lower band plus 30% of the difference between the two bands. As a filter, we're only going to place trades on Mondays and Wednesdays.

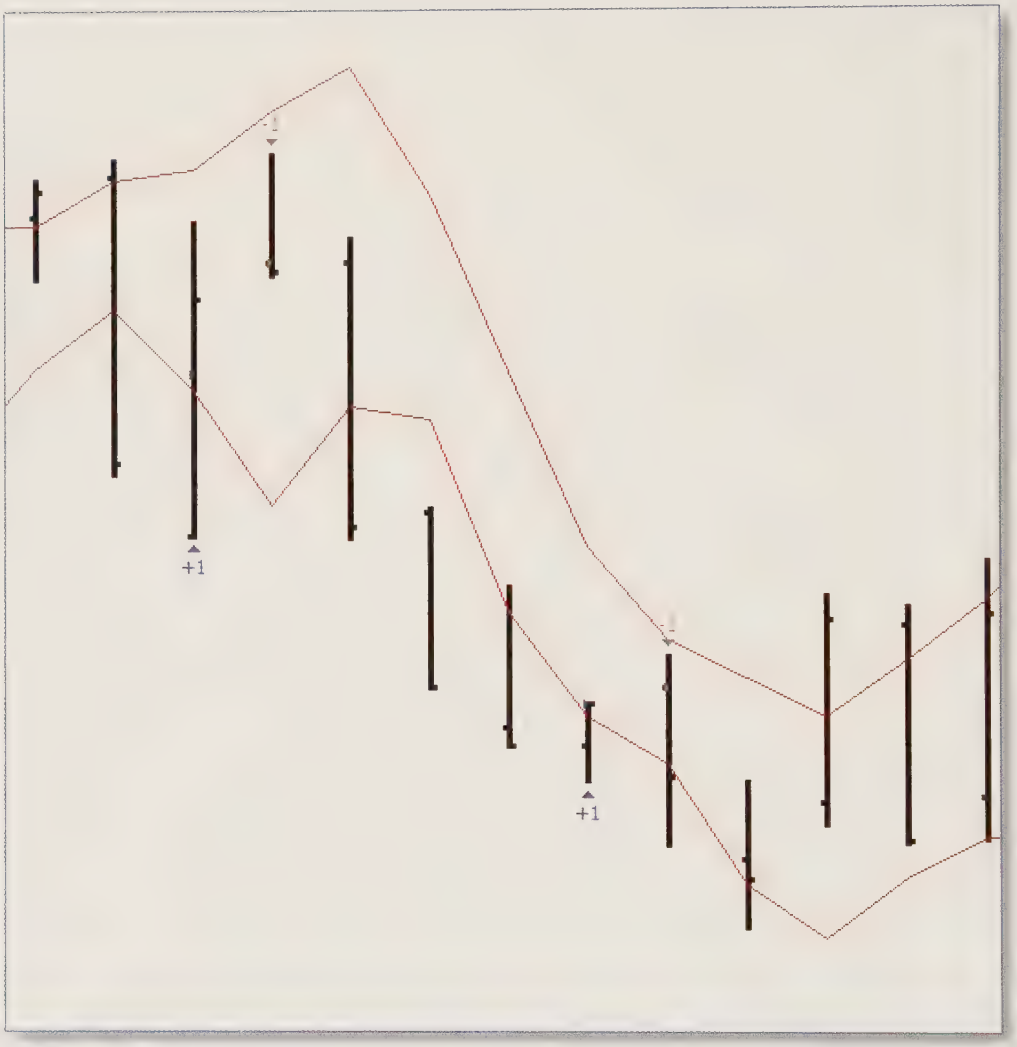


Figure 3.8: William Tell Bonds Pattern #12

This is typical trading band reversion-to-the-mean type of approach that attempts to identify when the market has gone too far towards an extreme. When the market opens outside these bands, it likes to try to get back inside them, especially on the days that Bonds tend to want to rally anyway. You can see two band trades in Figure 3.8, both taken during an overall down trend. Note how well this type of exit can still manage to generate profits, even if the market is running against it.

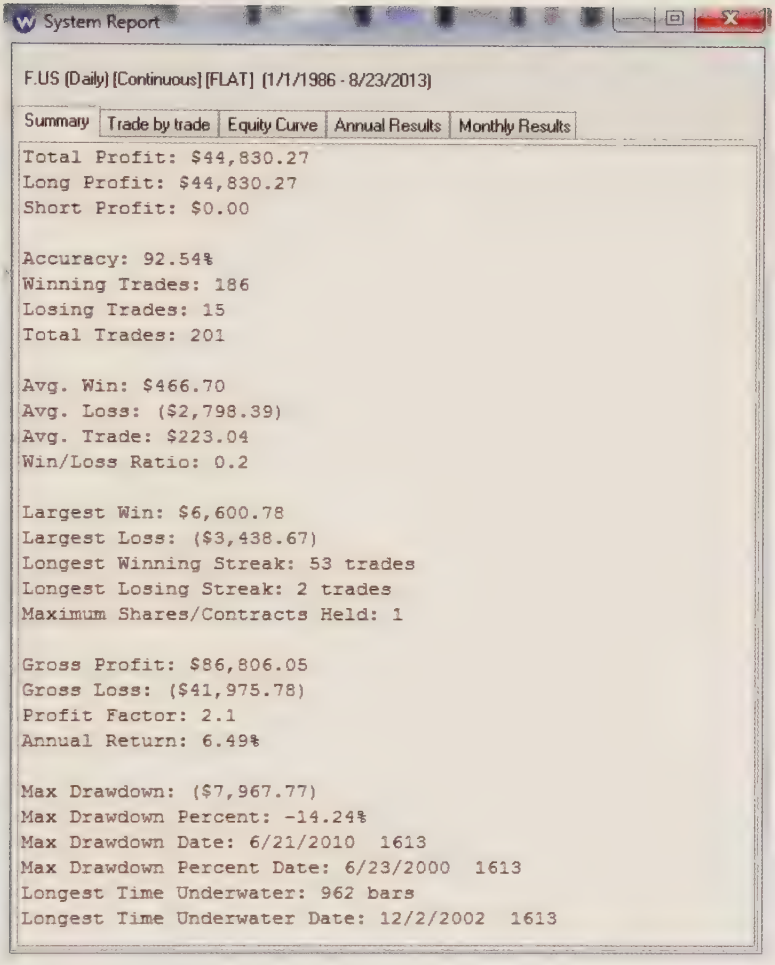


Figure 3.9: William Tell Bonds Pattern #12 System Report

The system report is shown above. We've lost a few percentage points in the accuracy department, but we are still sailing along at a sizzling 92.5%. Most importantly, check out the total trades. Over two hundred! That's ten times the number of trades we were looking at with the last pattern, and it puts this pattern in a league of its own. The idea that we might not have enough samples to judge the true effectiveness of this pattern is much less of a concern with this many trades, and now we're starting to see what will end up becoming the bread and butter trades behind the system.

Although back in the 90's, I tried to include all manner of patterns in my approach, if I had to re-develop the system today with the same goals, I'd shoot for two or three

patterns just like this one, and would forget all the rest. I just have much more respect for the importance of the number of trades, as well as being much more concerned about "brittle" systems.

Figure 3.10: William Tell Bonds Pattern #12 Equity Curve

This equity curve demonstrates how much more solid an active pattern is than an inactive one. Since there are so many trades, the pattern has a lot more time to dig itself out of holes. You can see this in both the 1999-2000 period, as well as the 2009-2010 period, in which the pattern took a couple losses in a row, but the equity curve isn't nearly as sensitive to them and is able to maintain a nice, upward trend.

Incidentally, this is the classic equity curve for a small win/large loss type system. It's a sort of saw-tooth pattern, where lots of wins result in a nice slow climb, but then large

losses give you the "teeth". Anytime you see an equity curve like this, you know that the system is running a similar sort of exit strategy to William Tell. Either that, or doubling-down on losers, as with martingale-style gambling systems. That style of trade management can also demonstrate the saw-tooth pattern, although the teeth are usually large enough to take the equity curve into the red zone in their case.

Coming back to my comment about brittle systems – that's a term I use to describe a system that is so reliant on a specific market condition or parameter set to perform that it blows up immediately when the market moves away from that parameter set being dominant. We'll talk more about parameters and optimization in another chapter, but in the case of these patterns, what we are really reliant upon is market volatility. It's very difficult for the market to move anywhere in a straight line, and markets like to do a stair-step pattern when they want to move in a specific direction. As long as our stop loss is far enough away, odds are that we'll get a profitable open somewhere before our stop is hit, simply because markets are noisy and have trouble being straight about their intentions. There's a reason the old timers say that you first take a market up in order to take it down.

Patterns like these will tend to work under two situations. First, if the pattern is actually based on something realistic, and isn't just random. And second, if the market volatility doesn't get so crazy that a stop of 2.75 points becomes easy to hit. The first issue would count as a failure in the development process, where a pattern made it into the system that really wasn't based on anything real. The second issue would count as a failure in assumptions about market character. Although I tried my hardest to avoid these sorts of situations, both of them have definitely crept into some of these patterns, especially as I sit here 16 years later looking over this material.

First let's look at pattern 16:

> *Pattern 16:*
> If we have a down close, compared to yesterday's close, and today's range is larger than yesterday's range, and tomorrow opens above today's close, then we're going to sell at a stop at today's low. Only trade this pattern on Friday.

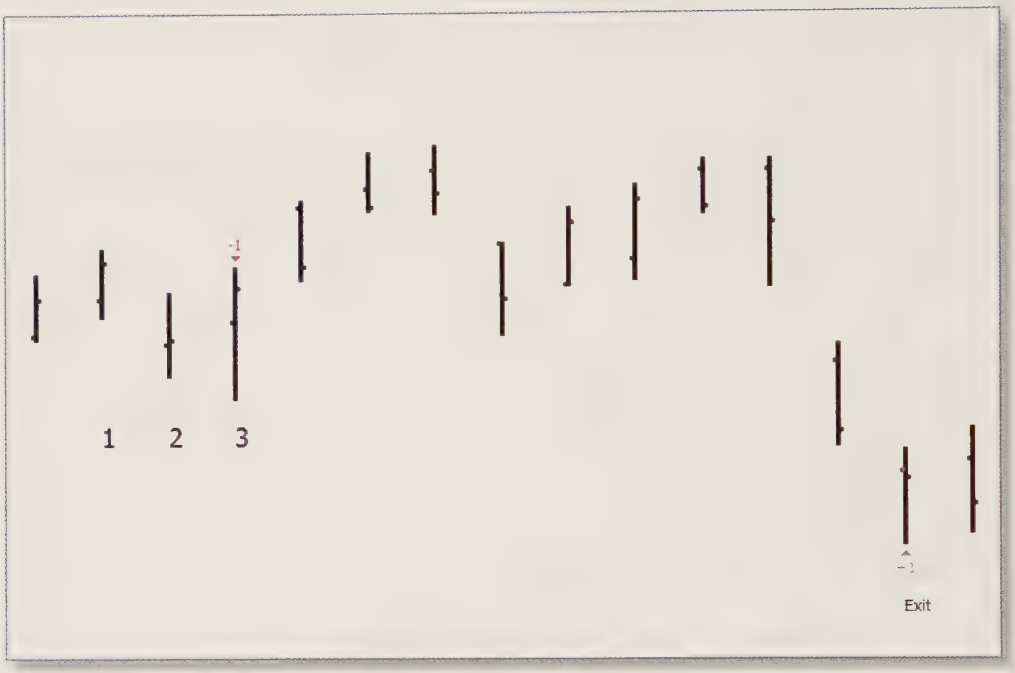


Figure 3.11: William Tell Bonds Pattern #16

Bar 2 on the chart is today's bar, and it is a lower close than yesterday (bar 1). It also has a larger range, although it's not so easy to see. We then sell at today's low on a stop, but only if tomorrow is Friday. Then get out at the first profitable open, which you can see towards the right edge of the chart.

I can see what I was trying to do – basically look for an expanding downward trend towards the weekend, to cause some panic that would push price lower on Monday. It must have worked out in my testing, as this pattern made it into the mix. But when we look at the equity curve, we see the true story:

Figure 3.12: William Tell Bonds Pattern #16 Equity Curve

This is what your equity curve will look like if you end up basing your patterns on smoke and mirrors. You can slice and dice market data a million different ways, and no matter how careful you are, you can still end up with conclusions that are simply false. In our case, expanding range downtrends towards the end of the week don't actually lead to Monday morning sell-offs. Bad developer! Incidentally, this system was still 86% accurate over this period. If you asked them, many beginning traders would jump at the chance to trade an 86% accurate pattern. You can still lose money trading futures, even if you're right almost nine times out of ten, and this is the proof.

Next, let's examine Pattern 18:

Pattern 18:

If today is an outside bar (higher high and lower low than yesterday) with a close higher than yesterday's, then sell tomorrow at the open minus 20% of today's range, on a stop. Only take the trade on Thursdays.

Outside bars and inside bars have attracted traders since the beginning of technical analysis, and this was my approach to trading them. I felt that they were fake-out moves (and still do!) and that fading them tended to work better than going with them in the short term. This system implemented that approach.

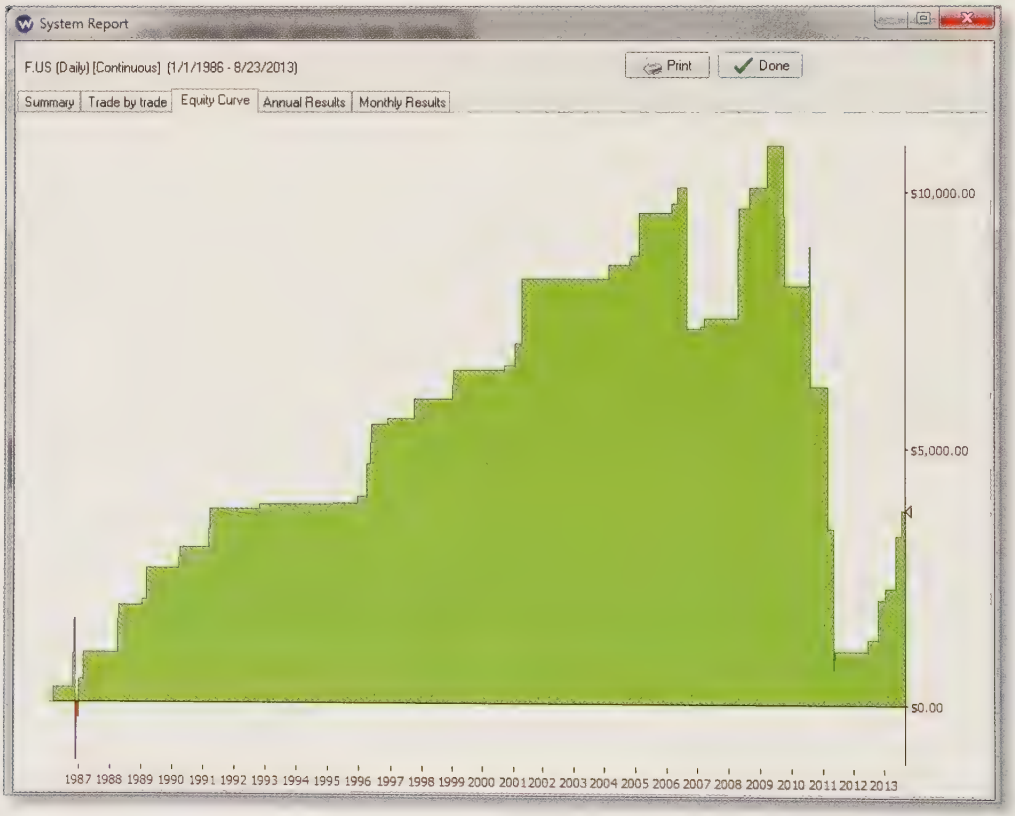


Figure 3.13: William Tell Bonds Pattern #18 Equity Curve

Can you see what happened? The pattern actually cranked away profitably for years, all the way up until 2010-2011, when it hit four losers in a row. What was happening during 2011 in the Bond market? Volatility for one, combined with a stock market that was

gyrating wildly, which created a nice bull market in bonds. Going short with large stops at that time wasn't the best thing to do. I'm not convinced that this pattern is busted, and if you take that one year out, everything looks pretty clean on the equity curve. Four losers in a short period of time is going to tank most of these patterns, and that's the Achilles' heel of this system. You'll see it occur when the bottom falls out of the financial markets.

Hopefully, the mechanics and ideas behind how to do this are all becoming clear. To be complete, here's a quick listing of each pattern and the general idea on which it is based. You can find the complete source code to this in the appendix so that you can take it apart and play with it to your heart's content if you're interested in experimenting with accuracy.

There are 19 patterns in the system. Each of them exits at the first profitable close, with a 2.75 point stop. Most of them employ some form of day-of-the-week filtering as well, which you'll also find in the code. Note that within Wave59's QScript, Monday is a "1", and each day counts forward from there in order, until you get to Sunday, which is a "7".

Pattern 1
Buy when the market closes beneath its low, and opens even lower the next day.

Pattern 2
Buy on an inside day with an up close, when tomorrow's open gaps above today's high.

Patterns 3, 4, 5, and 6
These patterns are based on one and two-bar Island Reversals, where an entire bar gaps up above or below the crowd, and then a future bar gaps back down. These were much more prominent back in the early days of this market, but only account for a handful of trades on the newer data set. I suspect the night market may have absorbed a lot of the energy that used to cause these patterns, but they are included for completeness sake.

Pattern 7

If today's bar gaps completely over yesterday's, then sell tomorrow if the gap is closed.

Pattern 8

This pattern tries to identify a 4-bar stair-step down pattern, and sells as the fourth bar takes out the low.

Pattern 9

This pattern identifies an upward trend, where lows are higher than the highs two bars back. If a pullback happens at the next open, a stop is placed above that open to get in on the move should it continue.

Pattern 10

This pattern identifies three down closes in a row (versus that bar's open), and when a pullback to the upside is detected, it goes short the next bar using a limit order above today's close.

Pattern 11

This pattern is based on indicators. It goes long when the PercentR oscillator crosses above an oversold threshold when a 5 period and 34 period moving average indicate an upward trend.

Pattern 12

This pattern looks to buy when the market opens under short term trading bands.

Pattern 13

This pattern tries to identify a pivot low where one low is surrounded on both sides by two higher lows. It buys on the break of a high after that pattern has formed.

Pattern 14

Similar to the last pattern, but only requires a pivot with one higher low on each side. Buys on a stop above the high.

Pattern 15

Looks to identify a breakaway gap, where today's low is above yesterday's high, and both bars are up. Buys on a limit at today's high.

Pattern 16

Looks for a downward close and expanding range to setup a short signal. If the next open pulls back, sell at a break of the low.

Pattern 17

Buy an outside bar with a down close.

Pattern 18

Sell an outside bar with an up close.

Pattern 19

This is a pure seasonal pattern. If it's a Monday in May, buy the close.

These nineteen patterns were then combined into one big system. The rules were to take whatever trade happened first, and to stick with it until it either got stopped out or exited at a profitable open. Then take whatever trade happened next. This would sometimes cause some confusion, as every now and then you have the chance to go either short or long on the same bar based on two different patterns firing together. Allowing reversals was another way to go about this, but resulted in worse performance, especially when considered from the all-important (at the time) accuracy metric.

As I'm working from my notes, I've built most of this back up within Wave59 for this book. The combined system does take multiple trades every now and then, but for the most part everything looks right so I'm just going to run with it at this point. Here's our combined equity curve:

Figure 3.14: William Tell Bonds Aggregate System Equity Curve

You can see that the system actually performed wonderfully, cranking away at almost 90% accuracy all the way up through to the 2008 market meltdown, when it slid into its worst-ever drawdown. At the time of this writing, market conditions have stabilized to the point where systems like this can be safely traded again, but it would have been a wild ride for anyone trading a version of this system a few years ago.

From my own perspective, this was the system I needed in order to trade myself out of the hole that I was in. I knew I was taking a risk due to the dangers associated with losses in this method, but at the same time, the odds were with me and I had nothing to lose. What a lot of people fail to realize is that trading systems have lifetimes associated with them. Some trading systems are designed for the 20-year long pull trader, and others are designed for the next few years. Trading systems tend to be like jobs or cars –

you use them for a few years, then move on to something else. By the time 2008 rolled around for me personally, I had already been through three or four other methods, and had long since gotten back into trading the ES in a discretionary fashion.

So for this particular system, what can we say about its lifetime? In my view, a system is considered to be successful as long as it maintains the structure of its equity curve from its development period. If you look at the last chart, and connect a line from the beginning of trading until live trading, then project it forward, you'll see that the system stuck with that line all the way up until the equity curve high in 2008. At that point, a new overall equity trendline developed that had a different slope.

To make it clear, I've drawn these three periods in on the next chart.

Figure 3.15: William Tell Bonds Aggregate System Equity Curve Lifetime

During development, equity curves tend to be their most perfect. The reason for this is that the developer has the ability to go back over the same data hundreds of times, looking for the methods that create the most ideal equity curves. In the case of William Tell, this is the line in black.

Eventually, the developer will begin trading the system with live money, on data that was unseen during development. If the developer has done a good job building the system, making sure that the inputs are based on real market forces and not magic numbers, then the system will generate profits for a period of time. This will continue up until some future period, when system degradation kicks in. The system's lifetime, as mentioned above, is until the period of degradation occurs. In our case, this is shown by the middle section (blue line) on the graph.

When judging the success or failure of a system, it is this middle area that is really used. There are two important considerations that need to be weighed when looking at this period with the idea of passing a judgment on how good a trading system was. First, how long did this profitable period last? Better systems have longer lifetimes than worse systems. Some poorly designed systems will fail immediately, and will never generate a nickel of profits for the unfortunate developer. In our case, the lifetime of our system was around 10 years, which is definitely in the "successful system" category. We'd like for our systems to continue profitably indefinitely, but that's not actually a realistic expectation. Remember that our system was developed using 17 years of data (1980-1997), so staying profitable for 10 years past that is a great accomplishment.

The next consideration that needs to be looked at is the slope of the equity curve in the middle section. Ideally, it should match the slope of the equity curve in the development period. In real life, that almost never happens, and the slope tends to flatten out. You can see that the blue line is just not as steep as the black one. That's because the black line uses the most optimal settings and parameters with the benefit of hindsight, while real trading utilizes educated guesses for these settings, which aren't always accurate. The result is a loss in the profit category, and usually a gain in the maximum drawdown category.

The third period of time is the degradation period, where the system has basically lost its edge. This happens to all systems eventually, for a number of reasons. First of all, as we'll see later, indicator parameter settings aren't stable, and move around over time.

Any time there's a reliance of any kind on fixed parameters, degradation will take place. Additionally, markets do change character. The Bond market of the 1970's and the Bond market of 2013 aren't the same. The big traders from the past have all been replaced by a different set of big traders, who use different tools and approaches. Economic conditions around the world change, different countries are in different relative positions on the global level, and even the market itself is different in terms of trading hours and contract specifications. A system developed for one period won't necessarily work the same way in another period. The more time that separates two periods, the higher the chance that system degradation will occur.

So what is a realistic expectation for system lifetime? A good rule of thumb is to take the length of the data set used to develop the system, then project forward by 40-50%. Properly designed systems will tend to remain profitable for that length of time. As an example, if we developed a system using 10 years of historical data, we should be able to get a system out of it that can continue to run for 4 to 5 years before degradation becomes a concern. This is a rough guideline - sometimes we'll get more time out of our system, and sometimes we'll get less. It's impossible to know until the degradation period has already taken place, which can take years to detect.

The solution to the issue of system lifetime is to rework the system periodically. You don't want to be constantly changing things around, as that is going to cause a whole other set of issues, but you need to schedule maintenance windows when you are allowed to update the rules to your system to keep is as current as possible. I remember reading an interview with a system developer that was very well respected back when I started trading, and he said that he would schedule these sorts of maintenance windows every 6 months on his intraday S&P system. Once he put a method into production, he was not allowed to change anything for six months, which included any sort of enhancements or new rules. Then, at the six month point, he would be allowed to make necessary changes, and another 6 month window would begin. By having a fixed schedule, you give yourself room to improve your method, but also prevent yourself from hurting the system by making emotional decisions during drawdowns. Tinkering is not always a good thing.

Had I continued trading Bonds using this type of approach, my equity curve would not have looked like that final one, as I never would have just sat on one system for a full 10 years without doing anything to it. Every few years, I'd have re-evaluated every one of

my patterns, especially in light of the growing database I'd have been collecting with current prices. Those patterns that continued to meet my criteria would have remained in play, and others that no longer continued to meet my criteria (such as the Island Reversal patterns), would have been retired. Additionally, more research would have been conducted, and more patterns would have been found and added to the collection, to create a similar, but different system to trade. In this way, adjustments for market character can be made, and systems can evolve over time.

More important than simply re-evaluating the patterns would have been the adjustments that I would have made due to skills that I had learned having developed more and more systems. Like anything, you get better at this the more you do it. Although I thought it was a good idea back in the 90's, I seriously doubt I'd use patterns based on day-of-the-week seasonal filters today, except in the most general of circumstances. Not that there's necessarily anything wrong with that approach, but frankly I just don't believe it's as stable of a method as I used to, and it feels to me now a bit like an attempt to over-optimize. I have a lot more experience these days, and my experience tells me to be careful with that sort of thing. So had I moved forward with this approach, other patterns would have been pruned that I felt were too dependent on that filter.

Before I close this chapter, I'd like to talk about a really important lesson that I learned not from trading this system, but from selling it. Once I had put this system into live trading, being an entrepreneurial person, I also thought it would be a good idea to sell the idea and let others use it. I figured it was 100x better than the Taylor method I had wasted my money on, and I had a good half year or so of live results to show, so why not leverage it and set up a second cash flow? I came up with a cool name, wrote a manual, and submitted the system to an independent system-tracking organization. This organization would track a bunch of systems submitted by developers, and would rank them each month in a publication which listed the best systems organized by market and by overall percentage gains. It was attractive to traders because all the results were based on out-of-sample data moving forward, which meant they weren't hypothetical, and they were collected by someone other than the developer, which helped in the credibility department.

Anyway, back then, you couldn't just submit *TradeStation* code for them to run, but had to completely disclose the method so this outfit could program it in their own software.

This lead to some issues for me, as I couldn't get them to match my results properly. They probably got it 80% correct, and I ended up letting them run with that, although I wasn't too happy about it. So even the independent testers don't have problems publishing numbers that they really know aren't totally correct. Welcome to the trading industry!

Despite those programming issues, in the first few months of trading, William Tell quickly hit #1 on the Bonds list, outperforming all the other systems in that market. It's actually really easy to do that when you're 100% accurate. Since most retail traders are concerned with accuracy above all else, people freaked out when they saw my accuracy figures and it wasn't long before I had a bunch of traders and brokers who wanted a piece of the action. I partnered with one of the brokers I met in Chicago, and traders would pay me quarterly to allow this broker to trade Bonds for them using this system.

Everything was great, and I had 30 people or so following the system. For a brand-new trader in his early 20's, it was exciting. People were doing well, and the equity curve was awesome. Then, we had a bit of a drawdown, when the system lost two in a row. You can see it on the chart, towards the beginning of 1999:

Figure 3.16: William Tell Bonds First Client Loss After Live Trading

I've circled the area in question. You can see the two losses in a row right at the start of that drawdown. Just to be clear, let's zoom in on that spot:

System Report
F.US (Daily) [Continuous] (1/1/1997 - 12/31/1999)
Print
Done
Summary
Trade by trade
Equity Curve
Annual Results
Monthly Results
$10,000.00
$5,000.00
1
2
2 losses = -$5,500
F M A M J J A S O N D 1998 F M A M J J A S O N D 1999 F M A M J J A S O N D

Figure 3.17: William Tell Bonds First Client Loss After Live Trading (zoomed in)

I've labeled both losing trades, which added up to a loss of $5,500 per contract. After the second loss, all my clients stopped trading the system! They couldn't handle the heat, and thought the system was self-destructing.

The problem was that although these people had read all the reports, they didn't really understand the system, and weren't prepared to trade it. They were going off of the latest fads in the trading magazine, and my system was doing well right then, so they wanted to jump in on it. They weren't looking at the markets from the perspective of the long run, but instead looked at them as if the immediate next trade was the only thing that mattered. Just compare the last two charts shown. The first one shows the long term history of this system, and the second shows only a couple years. The

drawdown period highlighted is exactly the same, but when looked at from the long term view, it's barely noticeable, and when looked at from the short term view, it looks really ugly. Perspective really matters with this sort of thing.

This is why most traders will never make any money in the markets. They are psychologically unprepared for drawdowns, and they take every loss personally. So they try one system, lose a few thousand dollars, and then jump to some other system. They stay on that other system for a few months, lose another few thousand dollars, then get scared and pull out, only to invest in some black-box method that again loses them a few more thousand dollars. Pretty soon, their accounts are blown, and they leave the markets in frustration, never to trade again.

If you are going to be able to make money in this business, you need to be able to weather some negative periods. It's part of the game, and there's no way around it. When trading this system (or any system), you need to be aware of what the worst case scenario looks like, and you need to be prepared for it. If the max drawdown number is around -$8,000 (as it was in this case), and you have decided to go ahead and trade this system, then why would you pull out after suffering a $5,000 loss? Is there any rational reason to pull the plug right then?

Unfortunately, most people are emotional, and trade emotionally. Trading emotionally has the negative consequence of locking in losing trades, as well as long-term negative returns. Yes, $5000 is a lot of money, but if you can't weather a $5,000 drawdown, you probably shouldn't be trading systems that have experienced drawdowns larger than that. In fact, you probably shouldn't be trading futures at all.

I knew most traders were undisciplined, but even so, I was very surprised to lose *ALL* of my clients after only two trades, especially after preparing them for the downside with all my reports. This experience really turned me off on the idea of sharing full-blown trading systems, which is why everything I've done publicly since then has been with discretionary approaches and not with mechanical systems. I just don't think the average person can handle systems that they haven't developed themselves, and I really don't like to see people lose money on my systems, especially if my systems are profitable!

So this book is breaking some new ground for me, in that I'm publicly venturing out into the trading system realm again. It's been awhile since I've been here. I don't expect human nature to have changed at all, but I'm also addressing a much smarter (and better looking!) audience than the one I had been working with before, so I'm optimistic that at least a few of you will be able to latch on to this work and use it profitably in your own trading.

This was quite a beefy chapter, and there's a lot in here. William Tell represents what every beginning trader really wants – lots of winning trades, and almost no losses. It does so a bit recklessly, as losing trades really hurt when they happen, but that's really the only way to go about getting the 90%+ accuracy numbers which this approach is really all about. The system as I've presented it needs to be updated a bit, but I don't see why, with a little work, it couldn't be adapted to work on modern markets. I'd start with Pattern 12, and see if you can find more like it. Three or four of those, and you'll be set. Just remember to take the long view, and don't get worked up over a losing trade here or there.

4. The Fallacy of Optimization

Once you start building and testing mechanical systems, it doesn't take very long until you run into the problem of parameter settings. Almost every technical indicator has some parameter setting that can be tweaked by the developer. For example, if we are going to calculate a moving average of the closes, we'll add up a certain number of closing prices, then divide by the number of days we aggregated to get the average closing value over the period. If we do this every day, we will have an indicator that is much smoother than the raw price value itself, which might be a useful input to our mechanical trading system. Unfortunately, once we start fooling with these averages, we'll discover that our trading system is profitable using certain length moving averages, but unprofitable using others. So how do we know which lengths to use?

Once computers came onto the scene, an easy solution was devised to try and solve this problem. Since computers are fast, they can compute profitability numbers for mechanical systems very easily. So what we can do is just have them compute all possible parameter values, and then select the best one. In the development world, this is called **optimization**.

As soon as optimization routines were added to charting software, legions of traders had the tools required to search through thousands of different combinations of indicators to find the ones that resulted in hugely profitable trading systems, with high accuracy and very low drawdown. If you have the tools to look at all the different possibilities, and the computing power to make it happen quickly, you can discover all sorts of interesting systems. Of course, the problem was that once these developers started trading these systems with real money, they found that the equity curves in their accounts tended to diverge significantly from the equity curves in their system reports.

The underlying issue here is curve-fitting. If you can go back over your data hundreds of thousands of times, of course you are going to find things that work. And the more filters, and tweaking, and adjusting you do, the more hypothetical money you can generate over that data set. The problem is that you don't actually get to trade that historical data with real money - all real trading takes place in the future, on a data set that is unknown.

The most glaring application of curve-fitting I've come across was something I read in an options trading book back when I was in college. This author had a strategy for how to go about selling options each month, and would make his money on time decay. An option is the right to buy or sell stock at a fixed price in a fixed amount of time. The option buyer gains the right to do this by paying the option seller a fee. As the option gets closer to expiration, its value declines, and eventually the option expires. If the stock hasn't risen or fallen the appropriate amount to put the option in the money, the option expires and is worthless. This is the ideal goal for traders who write options – they want to collect those fees, and then see the option expire with a value of zero.

Anyway, the strategy outlined in this book sold naked options, which is the same sort of thing as trading without a stop. You have no protection, so if you sell a put and give someone the right to sell their stock to you at a price of 100, and then the market drops to 50, now it's your job to suck up the difference, as they can make you buy their stock for 50 points more than it's worth. Options are (usually) traded in 100-share lots, so in this particular case, you'd lose $5000 on each options contract you had written. Since out-of-the-money options only trade for around $200 or $300 at best, it's a huge loss. This sort of options trading is a lot like the small-win/large-loss system approach we discussed last chapter, and only works if the option seller is very accurate.

What this guy would do is compute various standard deviation bands around price, then sell both puts and calls at the edge of the bands, hoping that price would stay between those bands. If it worked out, he collected premium on both the upside and downside. If the market moved a little, usually the combination of option time decay and the fact that he was collecting premium on both sides of the market saved him from a particular option going into the money near expiration. However, when testing his system going back in time, he had a problem with market crashes.

Selling naked put options right before the crash of 1987 resulted in massive losses. One wrong options trade put at this point in time resulted in cratering the entire account, rendering the entire approach a failure. Obviously, that's not a good thing, especially if you're writing a book about this particular method being the ultimate trading strategy ever invented. So what was this guy's solution? He added a simple rule to his system: no selling puts in October. Problem solved!

Do you see the issue here? To fix the system, this guy used knowledge of the past to create a filter that saved him in his backtesting. If your system misses that one huge trade that takes it out, then it's still a good system, right? He tried to rationalize this rule by claiming that it was a known fact that when stock markets crash, they do so in October, because that's when both the 1929 and 1987 crashes happened. So if you filter out October, you miss all the crashes. Remember our discussion of sample sizes and statistical significance? What can we say about a trading rule based on a sample size of 2?

Needless to say, had anyone been trading this method in 2001, 2008 or 2011, they are most likely not trading it now. Or, if they are, they will have had to make some additional deposits to their trading account in order to bring the balance up from $0. You can chop October out of all of those years, and still have enough downward movement to wipe a trading account out of existence pretty quickly.

This is the danger of building trading systems when you have knowledge of what happened in the past. You need to be very careful to do your testing in such a way that the system doesn't benefit from that knowledge. This is easier said than done, and sometimes just the act of optimization itself results in a curve-fit trading system, as you end up choosing a system that performed well based on nothing except the fact that the numbers worked during the period you are looking at. If that sort of approach was valid, we'd probably be working on lottery systems, as the bang for the buck is significantly larger than what we see in financial markets.

With that background, let's do an experiment, and see what we can learn about parameters and optimization in general. In this experiment, we're going to apply a moving average crossover system to the Dow Jones Industrial Average. This is an extremely simple approach: take two moving averages, and when the fast one crosses above the slow one, buy at the market. We then hold until the fast average crosses back under the slow average, and then we reverse and sell at the market. This system is always in the market, either long or short.

If you want to replicate this experiment on your own, or on a different market, here is the QScript source code I'm using to run this system:

```
input: short_length(5), long_length(10);

if  (barnum == barsback) bs=0;

if (short_length<long_length) {

  average1 = average(close,short_length);
  average2 = average(close,long_length);

  if (average1 crosses over average2) buy(1,close,"market","onebar");
  if (average1 crosses under average2) sell(1,close,"market","onebar");

  plot1 = average1;
  color1 = red;
  plot2 = average2;
  color2 = blue;

}
```

I'm going to use a simple moving average so the TradeStation users can play along, but you should be able to do this same thing with any other technical indicator.

What we will do next is apply this system to a 10-year period, and run a brute force optimization to find the best parameter set. For the fast average, we'll look at the averages with a length between 5 and 50, and for the slow average, we'll look at the averages with a length between 25 and 150. To speed the process up, I'll tell Wave59 to increment by steps of 5, meaning after checking all the length = 5 averages, then it will check length = 10, then length = 15, etc, all the way up to 50. It will do the same thing in 5 unit steps for the longer average. This gives us 260 combinations to check, which takes two or three minutes.

For the data set from 1995 through 2004, the best combination found was a fast average of 20, and a slow average of 30. The system report for this combination looks like this:

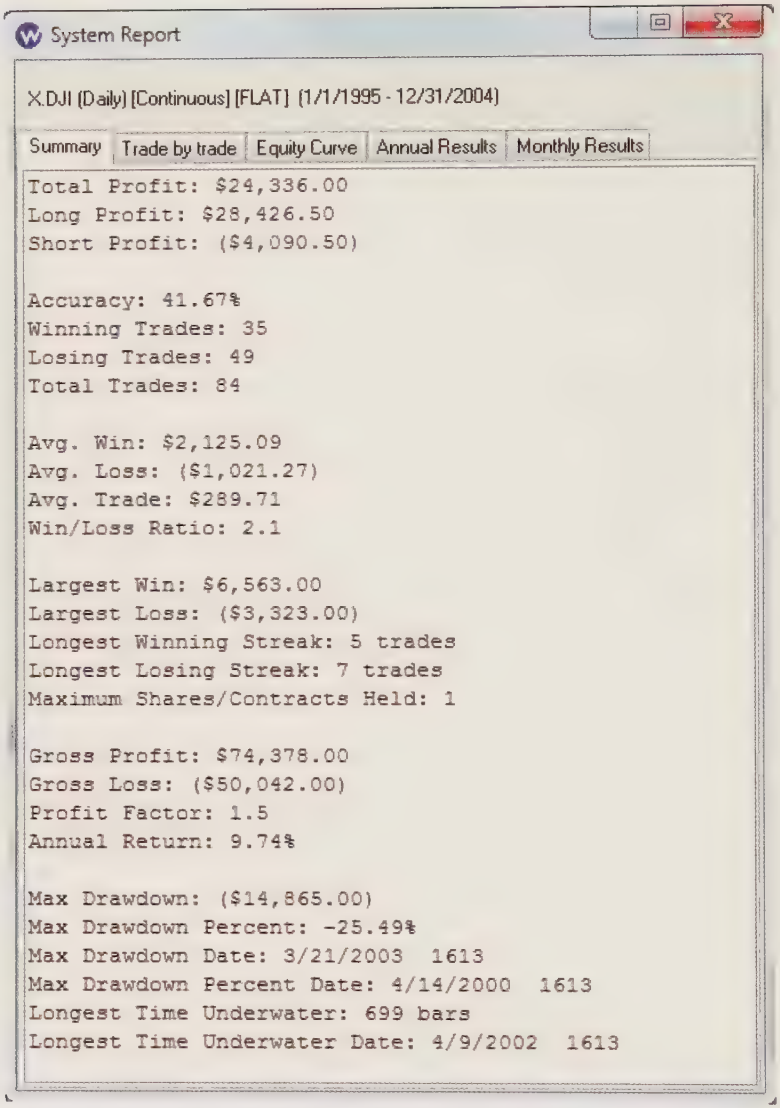


Figure 4.1: System Report for Optimized Moving Average System

Note that although we're running this on the raw Dow Jones Industrial index, testing was done as if we were trading Dow futures, which trade at $5 per point. The two are very similar, but the actual index has a lot more data available for testing, which is why we're using it even though trades would have to be placed on the futures.

This is a trend following system, and has about average accuracy for a trend follower at 42%. Notice the large win/loss ratio of 2.1. A little different from the William Tell patterns in that regard. Drawdown is at -$14,865, also about average for a trend follower.

Figure 4.2: Equity Curve for the MA System Optimized for 1995-2004

Here's the equity curve. Sort of the opposite of the high accuracy ones we've seen, as this one takes large steps up, and small steps down. It did have down periods, particularly around year 2000, but overall this is a decent equity curve given the simplicity of the system.

Let's move forward and apply this system to the next year, 2005, using the same parameters we have just optimized for...

Figure 4.3: Out of Sample Execution (2005) of MA System Optimized for 1995-2004

Ouch. Lots of red. Notice the nice slope on that equity curve. Too bad it's pointing down at 45 degrees rather than up! The method lost $6,474 in 2005, including the open trade at the end of the year. If we do the same optimization run in 2005, it turns out that the best parameters are 10 and 30, close to, but different enough from the 20/30 combination found on the previous 10 years to result in a negative return.

Let's move forward, and use a different data set, 1996-2005, and re-run the optimization. Doing this, we find that the 20/25 combination was the ideal, which made

$19k over the 10 year test period. Let's check that combination in 2006, the next year after our optimization period, as we did before.

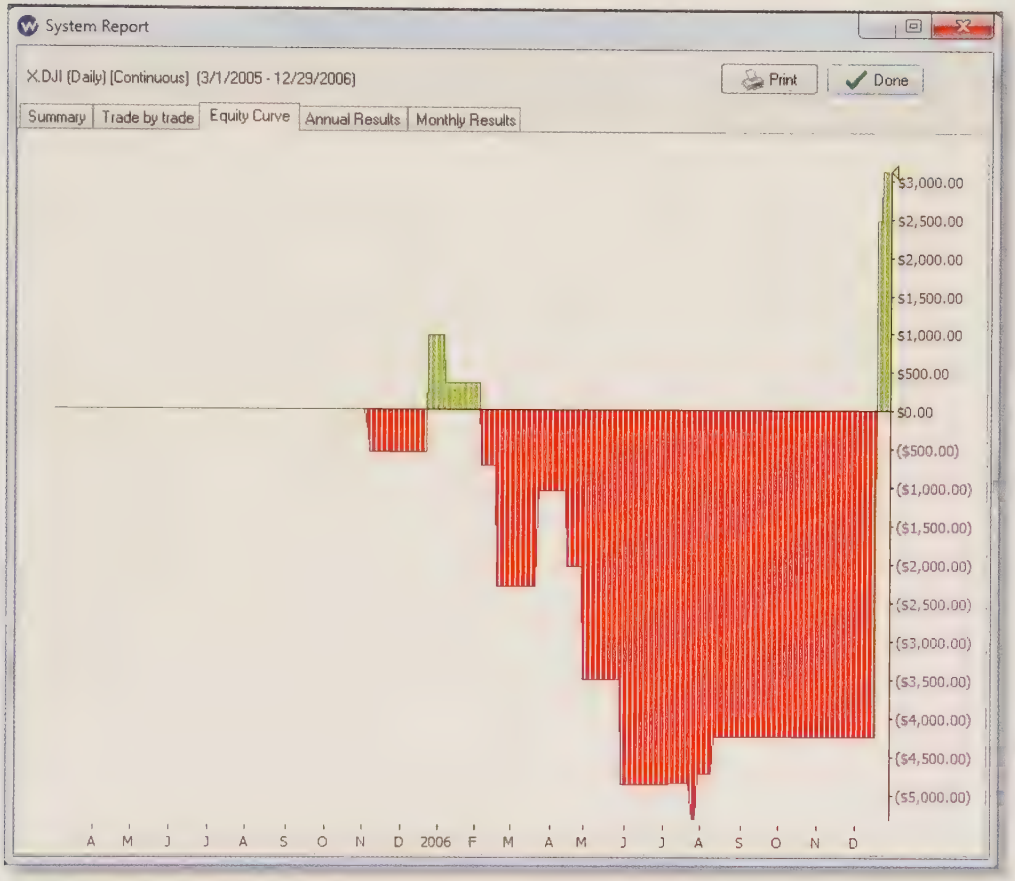


Figure 4.4: Out of Sample Execution (2006) of MA System Optimized for 1996-2005

Still lots of red, although we made it to positive territory in the nick of time to close out the year. The actual optimal parameter set during this period was 45/130, quite a bit different from 20/25, as well as from last period's 20/30.

Let's repeat this process starting all the way back in 1990, and move forward through 2012 (the last full year as of this writing), using the exact same procedure, and tabulate the results.

Optimization Period	Optimization Profit	Optimized Parameters	Out-of-Sample Year	Out-of-Sample Profit	Actual Best Parameters
1990-1999	$26,913	50/60	2000	-$17,825	45/120
1991-2000	$18,317	20/25	2001	$6,778	10/25
1992-2001	$26,135	20/25	2002	$1,550	30/90
1993-2002	$30,042	20/25	2003	-$1,218	30/35
1994-2003	$30,980	20/30	2004	-$7,848	50/75
1995-2004	$21,656	20/30	2005	-$6,474	10/30
1996-2005	$19,206	20/25	2006	$2,088	15/75
1997-2006	$10,984	20/25	2007	$4,125	45/130
1998-2007	$13,862	20/25	2008	$25,911	20/25
1999-2008	$38,972	20/25	2009	$14,076	20/25
2000-2009	$45,362	20/25	2010	$1,918	20/35
2001-2010	$35,510	20/25	2011	-$12,353	50/55
2002-2011	$34,140	50/85	2012	-$3,235	15/40

Table 4.1: P&L for Optimization Values over 10 Year Periods for MA System

One of the things that stood out to me when I first did a run like this was how frequently the yearly ideal parameters changed. In fact, each year was different than every other year except for the stretch in 2008-2009, where two 20/25 parameters repeated. This is quite a bit different than the 10-year optimizations, which tended to hover around 20/25 for most of the table.

If you think about it, that's somewhat of a surprising result. We'd expect that long term parameter sets would hold true looking forward in the immediate short term, but that's just not the case. In fact, except for the 2008-2009 period when the ideal parameters for both the 10-year period as well as the 1-year period were the same, those long term settings resulted in poor performance. Take either 2008 or 2009 out of the mix, and the entire approach turns into a loser over the whole test period.

In the last chapter we discussed the expected lifetime of a system, and I had mentioned a general rule that decent systems tended to stay profitable for about 50% of the length of time of the development period when moved forward into live trading. So if we developed on 10 years of data, we could expect the system to stay viable for roughly 5 more years. This observation does not apply to parameter optimizations, and, in this example at least, our parameter lifetime appears to be close to zero.

So what's the difference between William Tell patterns and these moving averages? The William Tell patterns are based on a very specific exit pattern, which is relatively robust. It can get blown out in bad periods, but if you run 100 different trials of it, most of those trials will result in a profitable outcome with a high accuracy. Therefore, as we move forward, we can expect the pattern to hold up, as there's something that makes sense about that exit strategy.

On the other hand, moving average crossovers don't have any similar sort of edge about them, and when we run them through an optimizer, we end up with "silver bullet" solutions. That's because if you take 100 different moving average trials, most of them will lose. So to make money with this approach, you have to get lucky, as we did in 2008 and 2009 when the 10-year optimizations happened to be the same as the 1-year optimal solution for that particular year. That won't usually be the case, and the usual result will be poor.

This is a critical point in system development. You need to make sure that your approach is based on something with an inherent edge. It's very easy to crunch numbers to produce historical profits when you are able to go over past data a bunch of times. If you can do that, you can take extremely unprofitable approaches, and still make them look good, because you can hand pick the parameter settings that just happened to work in the past.

Imagine that we have two different systems, System A and System B. Let's say we run System A over a 10 year period, using 250 different parameter settings, and find that 80% of those parameter settings produced profitable results, but the results were just average. Then, let's say we run System B over the same 10 year period, using 250 different parameter settings, and find that only 10% of the parameters resulted in profits, but those profits were much larger than the profits raised by System A. Which system should we trade, moving forward?

The answer is System A. The reason is that it has an actual edge in markets, which you know because it is indifferent to parameter settings. 80% of them worked, which means you have a very good chance of trading a profitable parameter set moving forward into the future. System B is one that you'd see for sale in a trading magazine. Although it made a lot of money in the past, it won't make anything in the future, because it is curve-fit, and has no edge. 90% of all of its parameters lost money, and since we know

that parameters tend to be unstable through time, we also know that the parameters used during development won't be the same ones needed during live trading. Therefore, the only way to make that sort of system work properly is to be able to guess at the "silver bullet" solution of the future that will happen to be profitable, and that's an impossible task.

Just to be thorough with this example, let's go back and do it again, but only optimize one year at a time, and run that setting forward into the next year. We can always argue that maybe the real parameters are changing too fast, and that a 10-year optimization window is just too long.

Optimization Period	Optimization Parameters	Out-of-Sample Period	Actual Best Out-of-Sample Parameters	Profit/Loss using Previous Optimal Parameters
1999	40/60	2000	45/120	-$18,891
2000	45/120	2001	10/25	-$13,673
2001	10/25	2002	30/90	$972
2002	30/90	2003	30/35	$4,553
2003	30/35	2004	50/75	-$8,348
2004	50/75	2005	10/30	-$6,988
2005	10/30	2006	15/75	-$4,065
2006	15/75	2007	45/130	-$5,868
2007	45/130	2008	20/25	$10,549
2008	20/25	2009	20/25	$14,076
2009	20/25	2010	20/35	$1,918
2010	20/35	2011	50/55	-$10,886
2011	50/55	2012	15/40	-$893

Table 4.2: P&L for Optimization Values over Single Year Period for MA System

The results are even worse than before, and this approach lead to a very negative result over the 12 year period. These optimal parameter sets are very slippery, and whether we optimize over a short period or a long period, they end up changing on us as we move forward into the future.

What we have been doing with optimizing a parameter set and rolling forward into a future period is the basis of a class of systems called **adaptive systems**. As a response to

the issue of unstable parameters, developers latched onto the idea of changing parameter sets on the fly in order to adapt to market conditions. I've had mixed luck with this approach.

The first issue with adaptive systems is that they can blow up in your face just as easily as systems that don't adapt. You can see this with our moving average crossover system above – we lost money by changing those moving average lengths each year, so in this case, getting fancy didn't help us. Adaptive systems will only work if the underlying approach has an edge. If it doesn't, you can adapt all you want, and you're still going to lose money. Simple moving average crossovers just don't work, and if you make them adaptive, that doesn't change anything.

The second issue with adaptive systems is that they are difficult to program. That's because your system actually needs to be a mini-optimizer, and run a bunch of different sub-systems at once in order to figure out which parameter set is working the best. Once it has an optimal parameter set, it can then use it to place trades on unseen data. If the system is simple, you can program something like this, but if it is complicated, it can be a huge challenge.

But since this is a book about mechanical systems, let's build one! The first thing we really need when approaching an adaptive system is some key edge. The purpose is NOT to find the "silver bullet" parameter to make a bad system work, but to let a system be fluid and not tied down to one particular parameter set. This means that the underlying structure of the system needs to be something that would work with lots of different parameters, not just one.

In our case, we've just been looking at moving average crossover systems, which have repeatedly blown up when we try to trade optimal parameter sets from one period into a future period. What about using that as our edge? I'm not talking about the moving averages, but the nature of markets to blow up when optimized. What we can do is curve-fit a system to the market, with the *expectation* that it will blow up as we go forward into the future period. So rather than trading those signals, we'll fade them!

Any repeatable feature found in markets counts as a possible edge to use when it comes to building a trading system. The key thing is that the edge is consistent through time, and gives us enough to generate a decent amount of profits. Markets have been

chewing moving average systems up since the beginning of time, and that's definitely something that isn't going to change. So why not take advantage of it?

From a programming standpoint, any adaptive system is going to need a key function whose job is to compute a profitability number for a particular parameter set. The system will then call this function a whole bunch of times with different parameter combinations, and will track which combinations perform the best over a particular period of time.

The full system can be found in the appendix, but the core function we are going to be using is also reproduced below.

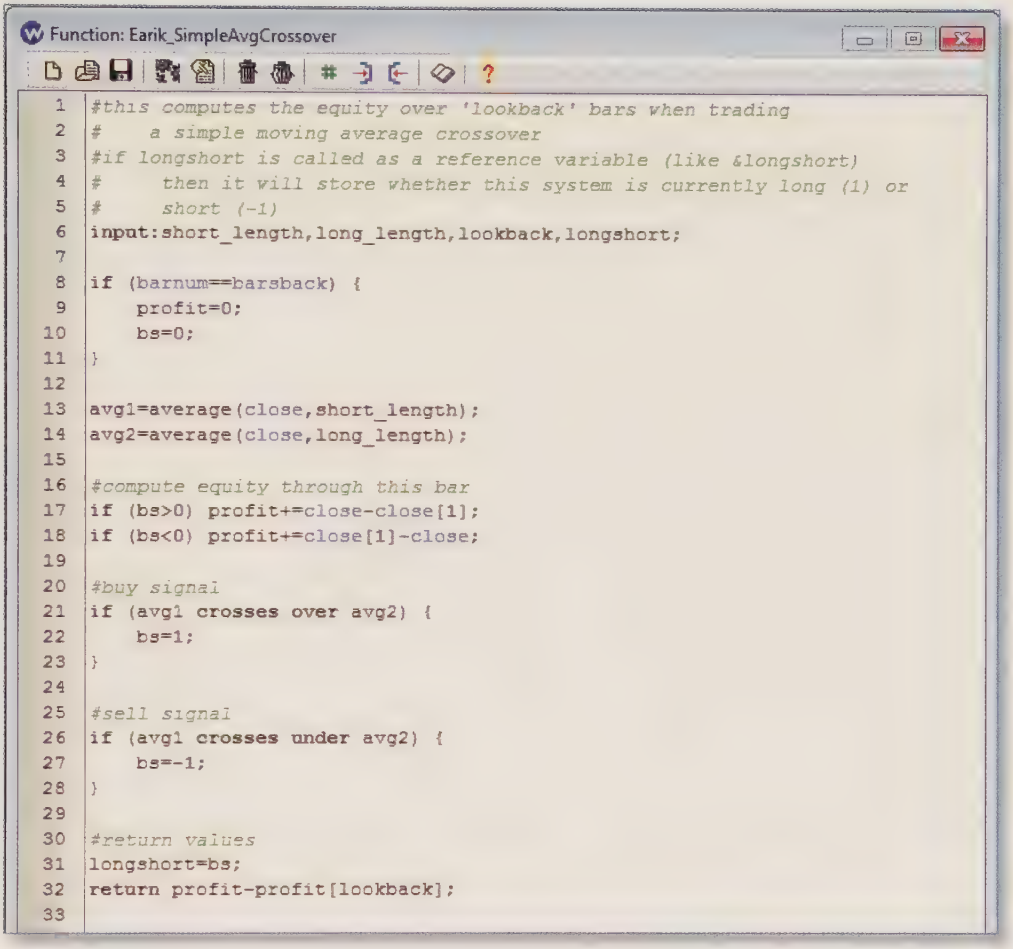


Figure 4.5: Adaptive System (main logic)

We will run this system and track all parameter combinations from 10-50 for the short average and from 20-100 for the long average. We will increment by steps of 10, which gives us 35 different average combinations. I'm doing this to save some computation time, but if I were going to run something like this live, I'd definitely figure a way to get more combinations in there. Since we want to find the most brittle solution we can, curve-fitting is a good thing, and you really do want to go crazy with it in this case.

Note that even in a system like this, we still have a parameter to worry about. Specifically, we need to decide what the *lookback* length should be in order to compute our equity values to determine the best system. In the hand-run examples we looked at

previously, we used a full year to compute the parameter combinations and then ran them on another full year. Computers can go much faster though, so I'm going to be doing this every single bar, for trades placed on the immediate next bar. But there is the issue of how long a period of time to use for tracking all those averages. This is where something like an optimizer really does come in handy, as we can test different combinations and see how they all performed.

Table 4.3 displays the results for the system placing trades from 2000 through 2012, the same dates as we used above.

Look-back Period (in bars)	Profit	Accuracy	Drawdown
25	$36,129	71%	-$25,546
50	$59,467	73%	-$16,773
75	$50,443	72%	-$19,155
100	$23,780	73%	-$23,595
125	$24,981	72%	-$19,504
150	$68,379	75%	-$19,018
175	$48,564	74%	-$22,475
200	$46,937	76%	-$19,155
225	$31,969	76%	-$26,166
250	$20,443	73%	-$29,952
275	$35,929	76%	-$26,297
300	$40,924	77%	-$29,349
325	$39,934	75%	-$30,896
350	$19,042	75%	-$35,943
375	$7,067	73%	-$38,339
400	$20,748	77%	-$40,440
425	$20,185	72%	-$35,598
450	$22,120	75%	-$31,556
475	$14,499	71%	-$33,417

Table 4.3: Adaptive System Optimization

When you run a brute force optimization, this is the sort of result that you want to see. Notice how the system made money on all of our selected parameters. Not all of them were great, and the best ones seem to cluster around 150 bars (half a year), but with a result like this you know that the parameters can move on you without turning the

system into a loser. What that means is that the base approach actually does have an edge.

Also notice the really high accuracy numbers. Turns out that moving averages can actually be both profitable and accurate, as long as you do the opposite! The numbers aren't good enough to actually trade live, but this would be a really good place to work on developing a system further. In particular, I'd consider a William Tell style exit approach here, as the signals themselves are already quite accurate.

The purpose of this chapter was to illustrate a very important feature of markets and mechanical systems. Parameters aren't stable at all. In fact, they are so unstable that you can create profitable systems based entirely on the instability itself, as we did with our adaptive system. Because of this, when developing systems, it is very important to keep an eye on those parameters. The ideal situation is to create a system that doesn't include any parameters to worry about, but that's not always practical. The next best situation is to have only a few parameters used by the system. The more parameters you have in your system, the higher the odds are that your system isn't going to be able to work well in the future. That's because all of those parameters will be moving around on you, and lots of parameters just gives Mr. Murphy that many more ways to turn the system on its head. This is why you'll find a very high correlation between good systems and simple ones. The more complicated you get, the more parameters tend to get involved in the process, and the less chance you have at actually coming up with something that will work in the future.

5. Mean Reversion Systems

In this chapter, we're going to take a look at a basic mean reversion system. Mean reversion systems are based on the idea that the market has a true value level which it tends to overshoot both to the upside and to the downside. These systems expect the market to try and revert back to its "correct" price, and make money by going short when the market is considered to be overbought, and going long when the market is considered to be oversold. Although I just used some terminology that tends to be applied to top and bottom picking, this class of system isn't a turning point picker.

There are many different types of mean reversion systems floating around, based on all sorts of different things. We'll actually get a chance to look at more than one of these in this book. One of the best ones I know about is based on the Balanced Centerline Indicator (BCI), which is something I developed years ago, back when Wave59 was still in its beta stages. Back then, wave59.com was a fledgling website, and I had a lot of questions about how to set a site up to be successful on the Internet. So what I decided to do was to implement a pop-up that would appear whenever anyone decided to leave the website, but only if they hadn't actually tried to buy anything. That way I got the opinions of people who got turned off by the site, or weren't interested in what I was doing. In the pop-up, I presented a quick survey, and told people that in exchange for their honest opinions I'd share the BCI with them as a sort of thank you.

The results of this pop-up were invaluable to me as feedback for how to go about improving the site, and had the added bonus of providing a really interesting look into the minds of the average retail trader. Although I shared this method with many people, only a few actually tried to do anything with it. Out of that group, maybe one person actually stuck with it for more than a couple months. I actually expected that, not because the BCI didn't work, but because it was so simple. One of the things you learn quickly after running a trading related business for a few years is that complicated things tend to be very popular, and simple ones tend to get ignored.

One of my good friends frequently asks me what I think about various ways to trade markets, and whenever I try and tell him about the best stuff, he always seems to respond with a comment like, "That's interesting, but if it were so simple, everyone would be doing it." Be very careful with preconceptions and biases, especially if you

aren't a profitable trader yet. Most of those ideas are completely false, and you'll have to unlearn them before you can be successful.

The BCI is a simple calculation designed to estimate the true value of a market at any given point in time, and can be used on any time frame the same way. The calculation is simply the square root of the high multiplied by the low. To put it in mathematical terms:

$$BCI = \sqrt{high\ x\ low}$$

This will result in a value close to (but not exactly at) the mid-point between the high and low of a given bar. It looks a lot like a fast moving average when plotted on a chart:

Figure 5.1: Balanced Center Line (BCI)

The free report that I had sent out detailed a few different ways to use this calculation, with the simplest being to watch what the market did at the next open in relation to the BCI value. Remember, this line represents an estimate of the price that the market wants to be traded at, so when you see price move above it, the usual reaction will be

for the market to drop. Similarly, when price falls below the BCI, the reaction tends to be a rally back to that level. One analogy that makes a lot of sense to me is to think about the market as running on a sheet of ice. It is trying very hard to get to a specific place, but it is always overshooting past its mark, and has to turn around and try to correct itself back towards the spot that it missed. Even though a market may be running in a strong trend, it will still try to correct this overshoot. Because of the nature of how this works, the BCI makes a perfect candidate for a mean reversion system. All we need is the BCI value, and a point in time against which to compare. The rules are:

Buy Signal:
If tomorrow's open is below the BCI, then buy at the market.

Sell Signal:
If tomorrow's open is above the BCI, then sell at the market.

And that's the entire system! You can compute the whole thing with a newspaper by hand, and don't need any charts, computer software, or data feeds. I told you it was simple.

I initially designed this for use on the big S&P futures contract, so let's take a look at that first, using the full data set, from April 21, 1982 all the way up to today, September 3, 2013. To keep this simple, I'm going to use a point multiplier of 250 for the entire period, even though this market traded at 500 per point during the first half of this period.

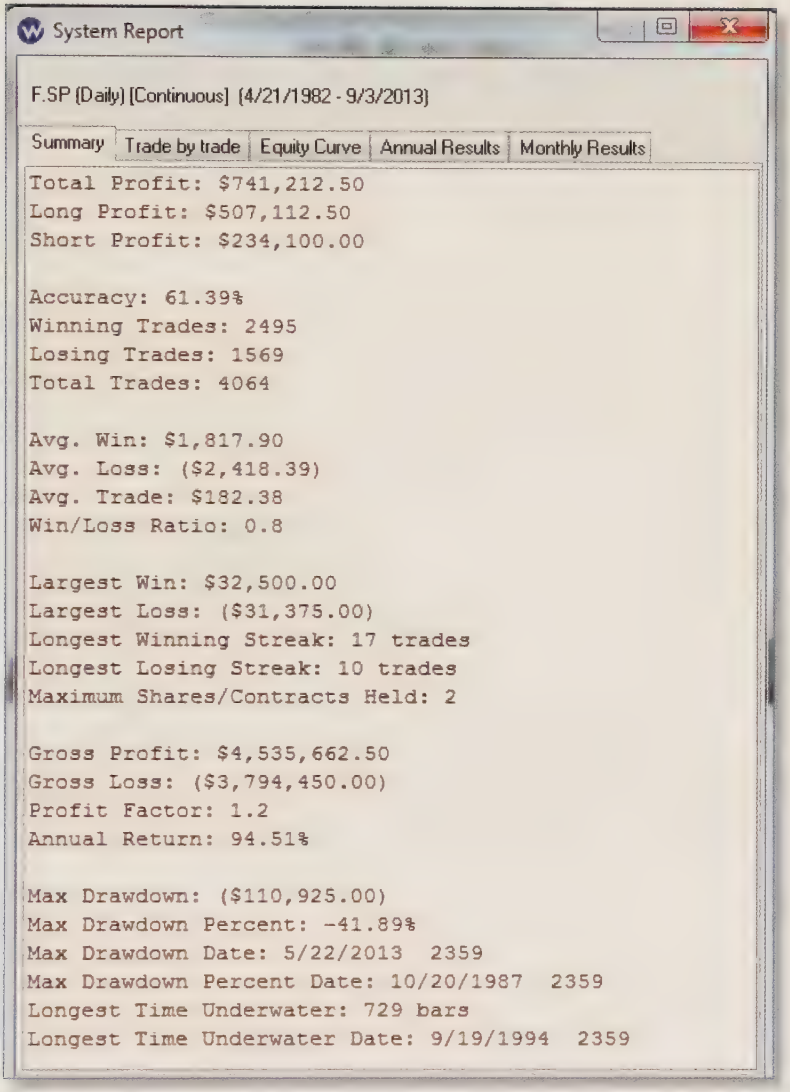


Figure 5.2: BCI System Report - S&P 500 Futures

What can we learn from this system report? First of all, this system made A LOT of money over the past 30 years. These results are based on trading just a single contract, and the approach has generated almost three-quarters of a million dollars in profits. Next, this system makes its money by being right. The 61% accuracy and win/loss ratio of 0.8 attest to that. It had a nasty drawdown in 2011 but this is a large point value market, so big profits tend to go hand-in-hand with big drawdowns.

Let's look at the equity curve:

Figure 5.3: BCI Equity Curve - S&P 500 Futures

This method does involve some sharp down-swings, but on the whole the equity curve is doing what we'd like and growing steadily. Notice in particular that almost half of this equity curve has taken place in unseen data after the system had been created.

Looking at the yearly statistics can give us more insight into the consistency of this method.

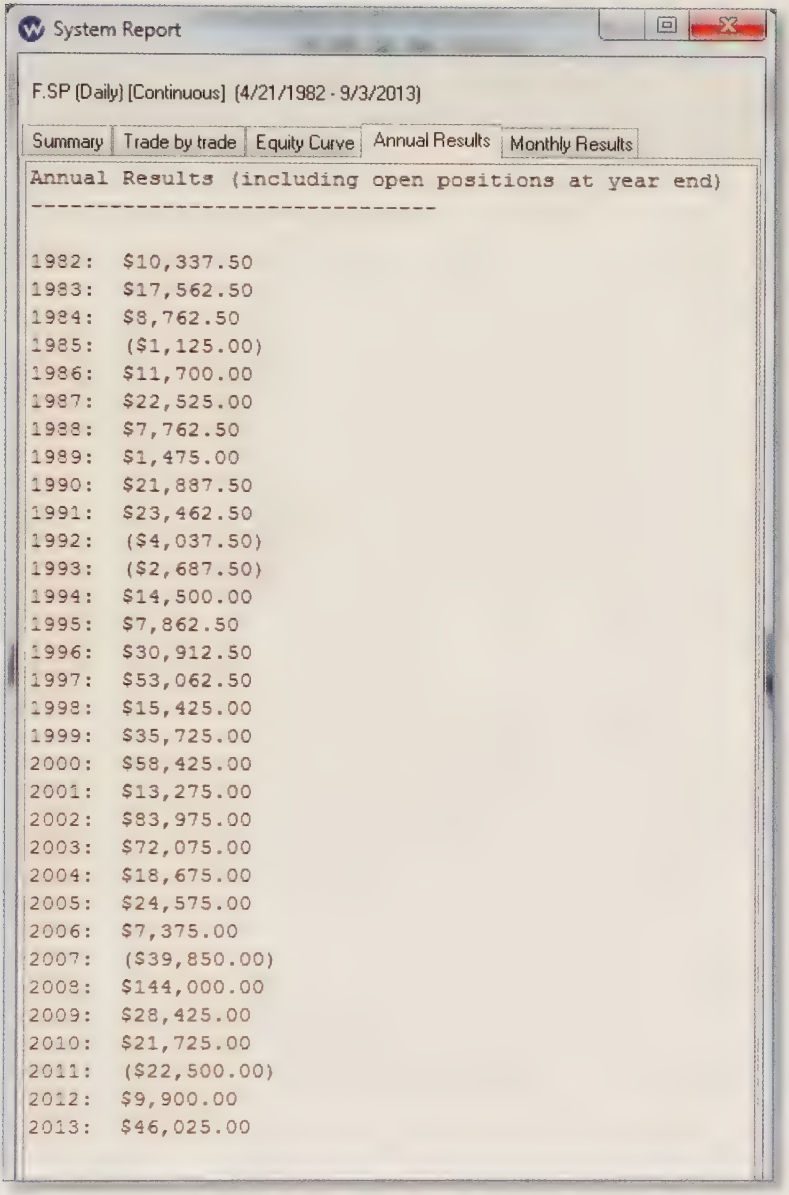


Figure 5.4: BCI Annual Results - S&P 500 Futures

Given thirty years of results, we find that the method made money in all but five of those years, and really only suffered two large losing years, one in 2007 and another in

2011. Also notice that it had some awesome years as well, especially in 2008 when it generated $144k per contract.

These are good results, and although a bit wild, this is tradable. In particular, take note of how many parameters this system uses: ZERO. There's no way to put this into an optimizer, because there is nothing to optimize! That's the ideal scenario, as it really reduces the chance that we've curve-fit these results. Out of all the systems I've had, this was the one that I always knew I could trade no matter what, without worrying too much about the system exploding on me. Yes, I had methods that produced much smoother equity curves, but I tend not to trust them as much. That's the enormous benefit of simplicity.

So what can we expect as far as system lifetime for something like this? There are no parameters, and no optimization, so there really hasn't been any curve fitting to unravel over time. This thing should keep on cranking away indefinitely, provided that the actual market itself doesn't change drastically in some way.

That last statement is actually important, because one of the key ingredients to what makes this work is that we need a market that has opening gaps. Remember that we're trading the overshoot past the ideal price, and we do so at the open. For a pit-traded market like the big S&P, this is a great approach, but for something like the E-mini, where the overshooting can get absorbed in the night session, you won't get as good of a result.

Although at one time, the big S&P was by far my favorite market, these days I don't think it's a great one to trade. It is nowhere near as fluid as the E-mini, and it's simply too huge for the average trader to be able to reasonably trade. There aren't too many individual traders who can absorb a -$100k drawdown at the single-contract level!

A good proxy would be the SPY, which is much friendlier to small accounts. It doesn't trade around the clock like the E-mini and so there are still pressure build-ups that make themselves felt at the open, which is what this system was designed to exploit. Let's take a look at that market as well, for completeness' sake. The SPY has 20 years of data available, which is plenty to work with.

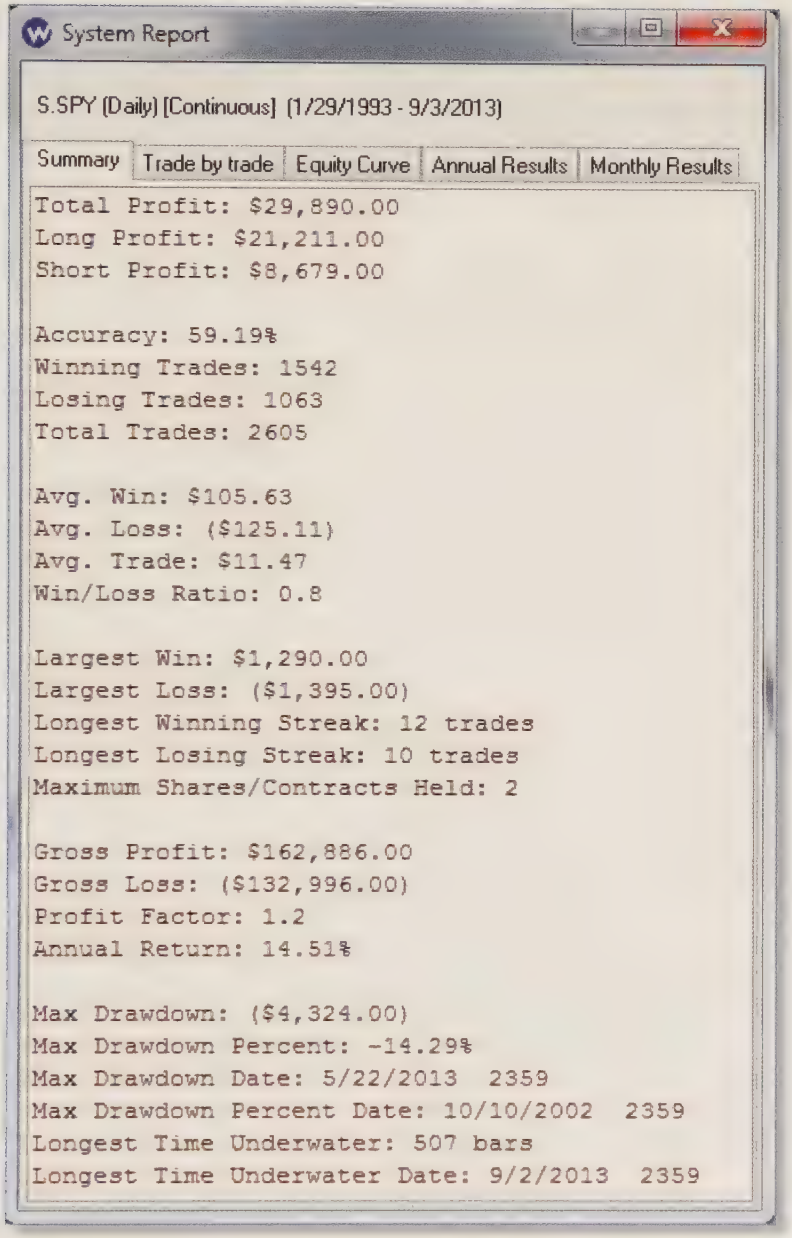


Figure 5.5: BCI System Report - SPY

These results are shown for trading a 100-lot block. The system is working well, although the average trade is somewhat small. If you consider slippage and commission

costs, you'll find that this is still very tradable. My broker provides one-cent (per share) commissions on a round turn, and the spread is usually only a cent, so we can say 3-4 cents per share for costs is what we'd expect. On a 100-lot, that's 3-4 dollars. Of course, options give you a whole lot of bang for your buck, especially with systems that trade this quickly since there's not a lot of time decay that eats away at positions, so that would be another way to do it.

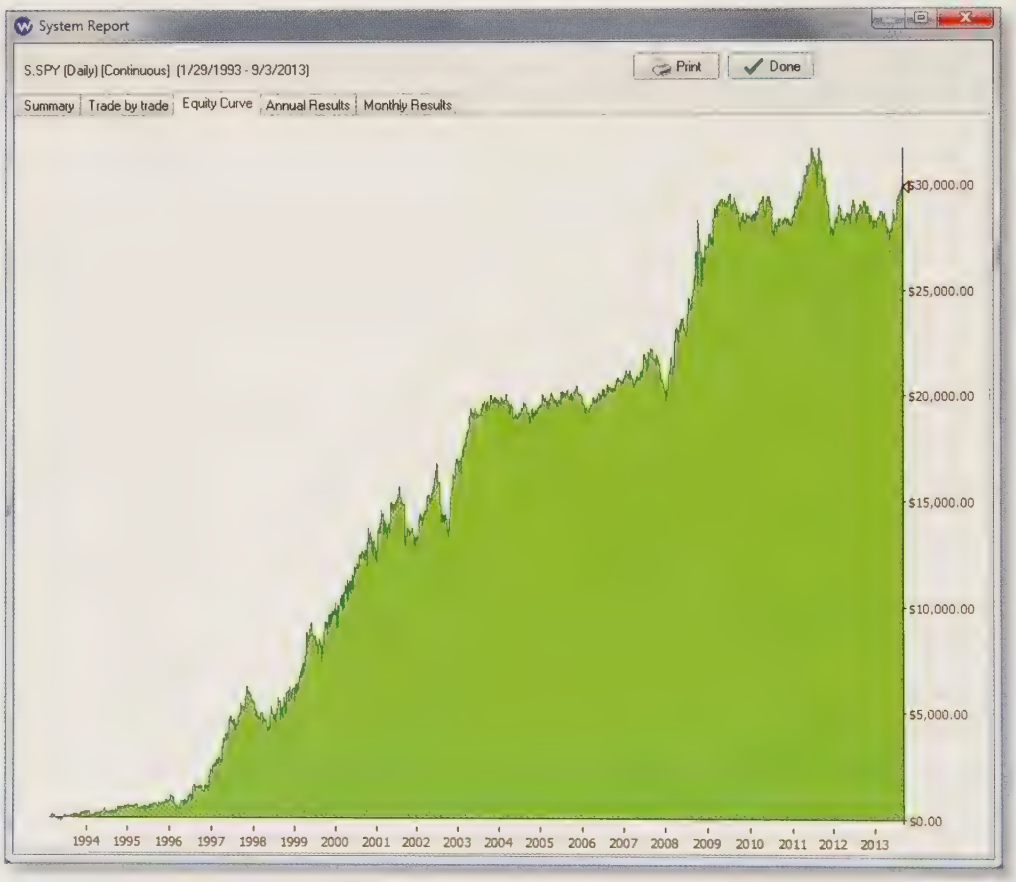


Figure 5.6: BCI Equity Curve - SPY

The equity curve looks very similar to what we saw with the big S&P futures. Again, the roughest period in the equity curve was the loss experienced in 2011, which the system is recovering from now.

This same approach works on most markets. Its weakness is that since it takes a lot of trades, and is always in the market, the average trade can end up being relatively small. So you have to make sure that your market can absorb slippage and commissions and still provide you a decent average trade to make this worth your while.

This is probably a good time to talk about one of the tricks that I've used when placing market orders via automated systems like this. Usually my systems do that right at the close, but the same method could be used at the open without any trouble. Rather than place a market order, what I do instead is place a limit order right at the price when the system has triggered a signal. In addition to this order, I instruct Wave59 to just take that order to market if it hasn't been filled within 15-45 seconds. What usually happens is that my limit order gets filled right away, just due to noise, and I save a tick on my entry using this strategy. Since I'm usually trading the ES, that's 0.25, or $12.50 per contract, which is a decent chunk when considered from the perspective of a whole bunch of trades. The times that I don't get filled on my limit order, I still get in via a market order, so I'm always able to get the trade as required by the system. All in all, the savings gained on the limit orders have outweighed the extra tick or two slippage on the market orders when price ended up running against me at the wrong time. Your mileage may vary with this, but I wanted to pass the technique along. For the programmers out there, the QScript function to check out in Wave59 is `IB_GoMarketAtTime`, which you can read about in the help files. The auto-trading template in the appendix uses this approach as well, so use that as a guide for how to implement this in your own systems.

So there you have it, your first mean reversion system. This class of system tends to exhibit a specific profile:

1) They tend to be reversal systems, always in the market.
2) They tend to make their money by being right. Usually the accuracy is 50-60% with win/loss ratios around 1.0.
3) They tend to like to buy and hold during market crashes, so things tend to get ugly during periods of extreme volatility. Of course, these systems don't have a monopoly on taking losses at those times either!

We'll look at more mean reversion systems later on, especially when we get into the machine learning chapter, so we'll get a chance to revisit this topic again. I've traded a

lot of mean reversion systems, and have probably made more money using them than all other styles combined. You need a strong stomach in order to handle weakness #3 as mentioned above, but aside from that, it's a great way to trade.

6. Stop Losses, Targets and Other Ways to Lose Money

Let me take a moment and demonstrate my awesome psychic abilities. After reading through the last chapter, you likely looked at the system report and equity curve, and thought to yourself, "This BCI system doesn't look half bad...! I bet if I worked on this and added some stops and targets to it, I could come up with a really good system."

Remember my comments in the last chapter about simplicity? What is the first thing that most traders do when presented with a robust, yet simple approach? They start adding things to it, trying to make it "better." In the process, the simple system turns into a complicated one, and a whole heaping of optimization and curve-fitting is thrown on top of a completely clean approach. It doesn't take long before a highly profitable system gets magically transformed into a complete loser under those conditions. "The Mona Lisa sure is a nice painting, but wouldn't it be better if we drew a Sharpie-moustache in on her? Too bad they've got that bulletproof glass there." Had we been talking about Art in the last chapter, and you had an equivalent thought, that would have been it - just so you know!

In this chapter, we're going to do some myth busting. For whatever reason, the trading world is one where there is a lot of misinformation flying around. Beginning traders show up, eager to learn about markets, and they spend all their money buying how-to trading books, which are filled will all sorts of notions about what works in markets and what doesn't.

On top of all that, people show up with very strong preconceptions about what makes markets work. Some of these beliefs are cultural, and we pick them up without even realizing it. Take a look at Astrology, for example. I know when I first came across the idea of using Astrology to time markets, I thought it was the stupidest thing I'd ever heard, and I thought the guy who had suggested it was off his rocker. Then I had to give it a day and think about it, and came back to the idea again, only to have the same reaction a second time. Of course, I had no proof that Astrology didn't work, and had never really looked into it before, let alone to the extent needed in order to confirm or deny anything I had just read. Instead, I was reacting emotionally, and was being totally superstitious about an idea that I just didn't like.

What happens is that our belief systems tell us that the world works in certain ways, and we go through our lives with a specific rule-set that tells us what is good and bad, or true and false. When it comes to markets, one of the things we are required to do in order to be successful traders is look at our ideas, and make two distinct piles, a pile for facts, and a pile for opinions. If you are honest with yourself, when you do this with all your beliefs about markets, you will find that most of the ideas end up in the opinion pile, and not the fact pile. For something to get labeled as a fact, you must have some proof to show that it is true. This goes not only for our beliefs about what works, but also about what doesn't work. Therefore, when you come across anything that you don't have any actual proof of, you must be careful about letting it slip into the fact pile simply because you like (or don't like) the idea.

Most people can't do this, and have trouble being really honest about their belief systems. This is also why most people lose money in the markets – their decisions are based on opinion, plus some emotion thrown in from their attachment to their money. The market moves around a little, gets them worked up, and the next thing you know they're closing out a losing position for no good reason. It's a feeding frenzy in there, and the retail trader is the chum.

One of the rules that I developed for myself early on was based on this observation. We know that most people are losing traders, and we also know that most people tend to have very similar beliefs about things. Therefore, if we want to be profitable, we have to learn to look for knowledge in the places where the average person's mind doesn't like to go. If everyone thinks something doesn't work, then it probably does! So what does that say about Astrology, the floofy pseudo-science that every Astronomy teacher likes to make fun of? Well, I'll be presenting an astro-based system for you next chapter! ☺ The flip-side of this is that when there's a rule that is blindly accepted among all traders, then it's probably wrong. Following the herd isn't going to get you anywhere you want to be in this industry.

The great thing about mechanical systems is that we have reports, and we test everything. If we have a method, we program it, and it either works or it doesn't. With a little practice, you'll find that you can test pretty much every idea you've got, and get numbers to either prove it or not. So once you've got that opinion pile all set out, you need to go through it and find out what is true, and what is false. Don't take anyone's word for anything, prove it all yourself, and do so with numbers.

With that in mind, let's run through some common beliefs about systems, and trading in general, and see how they stand up to some testing.

Myth #1: "Don't trade without a Stop"

What's the very first rule that you ever learned, that every trading educator in every introductory trading book tries to beat into your head? Don't trade without a stop-loss. Granted, if we were discretionary traders doing all of this by gut feel, we probably need to have a stop in there to help us when the market crashes and we lose our ability to think rationally. But that's not us, at least in this book. We are system developers, and we're building mechanical methods that don't require us to make decisions like that. These are automated rules that buy and sell based on hard logic.

Let's approach this completely logically. In order to add a stop-loss to our system, that stop-loss must do something that makes our system better than it was before. So let's take our BCI system from the last chapter, and add a stop to each trade. I'll set the stop value up as an input to the system, and we can then run it through the Optimizer in order to test different values.

I'm running this on the SPY, so I'll run through stops from 50 cents all the way up to $10 per share, in increments of 50 cents. That should cover the entire set of reasonable stop values that we'd really want to use in this case. For reference sake, the no-stop method generated $29,886 in profits over this same period with 59.26% accuracy.

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=0.5	$8,424.00	32.82%
2	#1#=1	$17,981.00	47.04%
3	#1#=1.5	$19,589.00	52.61%
4	#1#=2	$16,067.00	55.43%
5	#1#=2.5	$14,924.00	56.98%
6	#1#=3	$16,482.00	57.87%
7	#1#=3.5	$16,270.00	58.21%
8	#1#=4	$14,173.00	58.37%
9	#1#=4.5	$18,535.00	58.83%
10	#1#=5	$20,147.00	58.95%
11	#1#=5.5	$18,830.00	58.95%
12	#1#=6	$20,571.00	59.10%
13	#1#=6.5	$22,965.00	59.18%
14	#1#=7	$22,376.00	59.18%
15	#1#=7.5	$22,830.00	59.18%
16	#1#=8	$23,418.00	59.18%
17	#1#=8.5	$22,939.00	59.18%
18	#1#=9	$25,226.00	59.18%
19	#1#=9.5	$25,199.00	59.18%
20	#1#=10	$25,854.00	59.18%

Figure 6.1: BCI Stop Loss Optimization Test

Figure 6.1 shows the output from the Optimizer. The first column is the parameter setting that corresponds with the result shown in that row, which represents our stop value. The parameter being optimized is labeled as #1#, which is just Wave59's way of showing this. The next column is the profit or loss over the period, and the third column is the accuracy.

Notice the trend in this data – the tighter the stop is, the less money the system makes, and the less accurate it is. The wider the stop is, the more money the system makes, and the more accurate it is. Also, note that none of these cases was able to beat the no-stop run of $29,886 with 59.26% accuracy.

What is happening here? Specifically, stops serve one purpose – they lock losses in at a certain point. The more aggressive we are with locking those losses in (tight stops), the

worse our results become. Additionally, we can see that we are locking losses in from trades that ultimately would have ended up being profitable. That's why the accuracy is worse when the stops are tight.

So why would we want to use a stop-loss then? It doesn't improve system performance, and only hurts things. But don't we want to protect ourselves from the system going long during a market crash?

This system actually did buy and hold during 2011, which definitely caused a dip in the equity curve. Figure 6.2 shows the chart that depicts this trade.

Figure 6.2: BCI Example Outcome from Having No Stop

The system went long where labeled on the chart, and held during the drop all the way to the bottom, when it reversed just at the worst possible place. Had you been trading this system in 2011, you would not have been happy in August. So don't we want to avoid that then? The answer is no, we don't, and in fact, if we try to, we will lose money.

It comes down to this – you can avoid one big loss by using tight-stops, but what you will do is create many more small losses that end up adding up to a larger amount than that

one crash. So would you prefer to take your lump all at once, or would you prefer to double it, but spread it around and take a little bit every day? The point here is to make money, and we don't want to tie our system's hands when it comes to being profitable. Market crashes happen, and that's a fact of life. Luckily, they don't happen very often, and if we test properly we will know what to expect out of our method during those conditions.

This fear of market crashes is another instance of the ordinary trader getting caught up viewing the short term as being more important than the long term. To evaluate a system properly, you don't focus on one or two trades, but on a huge set of trades. We wouldn't trade a system based on only a handful of cherry-picked winners, and we shouldn't get worked up over a handful of cherry-picked losers either. We are looking at probabilities over a long period and this is a numbers game. Casinos do have to pay out jackpots from time to time, but you don't see them getting squeamish about it. They've run the numbers, and they know those jackpots are part of their business model. So they pay them out cheerfully, and let the profits take care of themselves. Be the Casino.

When we look at a system report, there are two metrics to use to judge how bad things can get in these worst-case scenarios. First, we have the Maximum Drawdown value, and next, we have the Largest Loss value. These are the two measures of negative shock to your trading account. Rather than focusing on each individual trade, it's better to focus on the entire account as a whole. The account will someday have to weather the storm described by those two risk metrics, so make sure that the losses incurred during that period don't hurt you too badly.

In the case of this system, the Max Drawdown was -$4,324 and the Largest Loss was -$1,395. Given that, you now have an idea how much you need in your account in order to trade 100 shares of this stock, using the approach described in Chapter 2. Figure out the maximum percentage loss you want to suffer in your trading campaign, and use these numbers to solve for the account size required to trade this within your safety parameters. Then, you can buy and hold through 2011. It will still be uncomfortable, but it won't wipe your account out, and the equity curve will bounce back in due course. Weathering drawdowns is something that has to be done one way or the other, so let your real stop-loss be found in your position sizing, not in your individual trade management.

What you'll find as you test more and more systems is that stop-losses almost invariably degrade system performance. Sometimes you need them, as in the case of William Tell, because the system might not be able to exit otherwise, but for reversal systems that manage their own exits, I can't think of a single time I've ever seen stops improve the numbers. And if the numbers don't support the use of stops, then why exactly would we be using them?

There's only one situation that I use a stop with, and that's the catastrophe stop. The **catastrophe stop** is a special case stop, designed to stop a system from trading in case of exceptional circumstances never before seen. It should never (or almost never) have been hit in past testing, and its only goal is to let the system live to fight another day in the face of overwhelming global market meltdown due to alien invasion or the pending zombie apocalypse. I've got a reversal system I trade that has a catastrophe stop set to pull the plug should the account lose 70% of its value in one trade. Historical testing has never even come close to a fraction of that loss, but that level is maintained just in case, to at least give the system a chance to recover rather than being completely wiped out by a rogue wave. I don't expect it ever to come close to being hit, but it's there anyway, more for psychological reasons than anything else. Practically speaking, it's basically the same as not having one.

Myth #2: "Profit Targets are a Good Thing", or, more popularly, "No One Ever Went Broke Taking a Profit"

This is the reverse of using stop-losses. Rather than trying to limit a loss, using profit targets attempts to maximize winners. Let's run the same test as we did before, using the BCI system on the SPY. I'll tell the system to exit all the way from 50 cents profit up to $10 of profit, in steps of 50 cents. Everything else remains the same.

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=0.5	$14,491.00	71.45%
2	#1#=1	$22,204.00	63.91%
3	#1#=1.5	$22,663.00	60.42%
4	#1#=2	$24,440.00	59.64%
5	#1#=2.5	$26,054.00	59.57%
6	#1#=3	$27,043.00	59.30%
7	#1#=3.5	$26,968.00	59.26%
8	#1#=4	$27,160.00	59.26%
9	#1#=4.5	$27,043.00	59.18%
10	#1#=5	$27,923.00	59.18%
11	#1#=5.5	$28,774.00	59.26%
12	#1#=6	$29,117.00	59.26%
13	#1#=6.5	$29,077.00	59.26%
14	#1#=7	$29,677.00	59.26%
15	#1#=7.5	$29,650.00	59.26%
16	#1#=8	$29,492.00	59.26%
17	#1#=8.5	$29,643.00	59.30%
18	#1#=9	$29,705.00	59.26%
19	#1#=9.5	$29,603.00	59.26%
20	#1#=10	$29,703.00	59.26%

Figure 6.3: BCI Profit Target Optimization Test

Again the first column is the profit target value, the second is the total profit, and the third is the accuracy. Like with the previous result, the tighter the profit target, the less profit the system makes overall, and the larger the profit target, the more the system makes. The accuracy numbers are reversed from before, with the tightest targets having the highest accuracy. That makes sense, because the system is able to turn some losers into winners with a tight target. Again, although some of the larger targets came close, no solution is able to beat our original gain of $29,886.

Why don't limits outperform the original? The reason is that the original took some really large winning trades. If you consult the original report, you'll see that the largest winning trade ever was +$1,290. When you start adding targets, what you do is force those big wins to be small ones, and profit drops. As it turns out, it's worse to cut those

big winners short than it is to turn some of the losers into small wins, and the numbers clearly show that. So while stop-losses result in locking in losing trades, profit targets result in limiting winning ones. In both cases, overall performance is harmed.

This reminds me of an exit strategy that was made popular in one of the trading psychology books, which I've seen repeated over and over, not only in the general trading arena, but also within the Wave59 community itself by traders that I would normally expect to know better. The technique requires that you put two contracts on for each trade, then, as the trade moves in your favor, you take a quick profit on one contract to "lock in" a winning trade, and let the second contract run for an attempt at a large gain. Sometimes a third contract is added with a secondary profit take at a larger amount. The technique makes intuitive sense, as it allows otherwise losing trades to be quickly brought to break-even, improving overall accuracy. Traders also like to ring the cash register, and taking a quick profit right away is every beginning trader's instinctual behavior. Cash that winner in while the money is on the table!

But since we have testing tools, when it comes time to evaluate this sort of thing, we can do so objectively. Anytime you see a system with multiple contracts and multiple exits, just realize that there are actually multiple systems being traded together. So in the case of the quick exit and the slow exit approach, what we would do is test both exits separately. Then, after evaluating each system independently, see whether it makes sense to trade that particular system on its own.

In the case of the quick profit take, that's basically just trading a system with a very tight target. If you look at the last optimization run we just did on the BCI (Figure 6.3), that would be the first line, where the system generated $14,491 with 71% accuracy. Yes, it's accurate, but it's only half as profitable as trading the regular way.

Aficionados of this approach tend to acknowledge the fact that, on its own, the first profit take doesn't look all that great, but they contend that the combination of the two is what matters, and they accept smaller profits on the first trade in order to smooth out the portfolio equity as a whole. My take is that they are focusing on the short term to the detriment of the long term, and probably haven't actually run any numbers. Look at it this way – if you are going to be trading multiple units, then why not trade all the units in the most optimal way? So rather than taking one off quickly to try for higher accuracy, why not trade two in the regular way, letting both run for their maximum

gain? It's not like the quick exit is risk-free. There's associated drawdown added by having an additional contract in the trade, and if you are going to take risk, why not maximize the gain?

I guarantee that if you look at the quick exit approach next to simply trading two contracts using the regular system rules, the equity curve from the full 2-contract system will outperform the equity curve from the quick exit approach, usually by a significant margin. Yes, in the short run, when focusing on one or two selected trades, the quick exit method will appear to add some value. But when focusing on the long run, and looking at annual percentage returns, the real story will emerge. If you are going to trade, trade to make money, and don't get caught up in the moment.

But, unfortunately a big name psychology author wrote about this exit, and now everyone believes it's a good method because they like the idea. Don't take anything for granted. Prove it before you use it, and don't let useless trading lore find its way into your fact pile without legitimately earning that position.

Myth #3: "I Can Improve this System by Adding Stops and Limits"

We have already pretty much dealt with this, but it's good to be thorough. So let's create a system that uses both stops and limits, just like good system developers are taught. I'll run the tests from 50 cents to $10 again, in steps of 50 cents, for both the stop and target values. That comes out to 400 combinations, so I'm not going to show them all, but instead I will focus on the top-20 and bottom-20.

First we'll look at the bottom-20, the worst results of the entire run:

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=0.5,#2#=0.5	$1,740.00	49.46%
2	#1#=0.5,#2#=2.5	$5,160.00	70.83%
3	#1#=0.5,#2#=2	$5,349.00	69.82%
4	#1#=0.5,#2#=1	$5,732.00	63.56%
5	#1#=1.5,#2#=0.5	$6,219.00	35.37%
6	#1#=1,#2#=0.5	$6,782.00	39.72%
7	#1#=0.5,#2#=4	$7,059.00	71.68%
8	#1#=3,#2#=0.5	$7,266.00	33.75%
9	#1#=0.5,#2#=1.5	$7,308.00	67.85%
10	#1#=0.5,#2#=3	$7,343.00	71.29%
11	#1#=2.5,#2#=0.5	$7,379.00	33.82%
12	#1#=0.5,#2#=3.5	$7,456.00	71.56%
13	#1#=2,#2#=0.5	$7,666.00	34.40%
14	#1#=1.5,#2#=4	$7,827.00	60.38%
15	#1#=3.5,#2#=0.5	$8,123.00	33.59%
16	#1#=4,#2#=0.5	$8,202.00	33.55%
17	#1#=4.5,#2#=0.5	$8,255.00	33.44%
18	#1#=5,#2#=0.5	$8,436.00	33.36%
19	#1#=5.5,#2#=0.5	$8,584.00	33.40%
20	#1#=6.5,#2#=0.5	$8,590.00	33.44%
21	#1#=6,#2#=0.5	$8,594.00	33.48%

Figure 6.4: BCI SL and PT Optimization Run Worst 20

The first parameter is the profit target value (#1#), and the second is the stop-loss value (#2#). You can see that the absolute worst thing to do is to use both a tight target and tight stop at the same time. In this case, our system is basically flat, earning $1,740 in 20 years, not counting commissions. Ouch!

The numbers go up as things loosen, as you can see in the top-20:

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=10,#2#=10	$27,481.00	59.18%
2	#1#=9.5,#2#=10	$27,431.00	59.18%
3	#1#=10,#2#=9.5	$27,074.00	59.22%
4	#1#=9,#2#=10	$26,992.00	59.18%
5	#1#=10,#2#=9	$26,943.00	59.18%
6	#1#=9.5,#2#=9	$26,893.00	59.18%
7	#1#=8.5,#2#=10	$26,892.00	59.18%
8	#1#=7.5,#2#=10	$26,888.00	59.18%
9	#1#=8,#2#=10	$26,792.00	59.18%
10	#1#=9.5,#2#=9.5	$26,726.00	59.18%
11	#1#=7,#2#=10	$26,445.00	59.18%
12	#1#=9,#2#=9.5	$26,287.00	59.18%
13	#1#=9,#2#=9	$26,206.00	59.14%
14	#1#=8.5,#2#=9.5	$26,187.00	59.18%
15	#1#=7.5,#2#=9.5	$26,183.00	59.18%
16	#1#=6.5,#2#=10	$26,145.00	59.18%
17	#1#=8,#2#=9.5	$26,087.00	59.18%
18	#1#=7,#2#=9.5	$26,038.00	59.22%
19	#1#=6,#2#=10	$25,845.00	59.18%
20	#1#=6.5,#2#=9.5	$25,738.00	59.22%
21	#1#=8.5,#2#=9	$25,706.00	59.10%

Figure 6.5: BCI SL and PT Optimization Run Best 20

It should be no surprise that the best solution is the one where the stop and limit values are set to the maximum allowed for the run, at $10 each. As they tighten up, profitability drops off. More to the point, in 400 tests, not a single stop/target combination was able to improve upon the original raw reversal system at $29,886.

So you might be asking yourself, "Am I really supposed to trade without stops?" No, that's not what I'm saying. At least, not exactly... My point is that whenever you add or modify something, make sure that doing so actually improves the system and doesn't hurt it. Don't do anything on the basis of dogma, or because it *seems* like a good idea. Do it because you can prove that the method got better for the addition, and to prove that, you need to be able to say exactly how much better, in terms of dollars and cents.

When I first started trading, I had a ton of biases and incorrect ideas about markets. The only way I was able to sort through it all was to test each idea separately. The numbers can be made to lie, no doubt, but if you do lots of testing, you'll start seeing patterns, and the general rules will start making themselves known. For example, if every time you try to add stops to your system, you find that your system degrades, that is telling you something. Most of what you'll find to work in markets completely flies in the face of generally accepted trading principles, and being successful is going to require that you throw a lot of those principles out the window. Trading without a stop doesn't mean that you are taking on unlimited risk. Quite the contrary, you no longer have the crutch of a stop to lean on, so you really have to figure out the correct account size to use when applying your system to maintain your personal risk profile. Trading is all about managing risk, and part of that is removing the things that cause you to lose money. If you can show, with numbers, that stop-losses actually cause you to lose money, then you really have no choice but to remove them from your methods. But don't let that scare you, there are other methods to kill off a losing trade that don't have the nasty side effect of tanking your results, which we'll look at before this is all said and done.

7. Lunar System

In this chapter, we're going to build a system from the ground up, starting with our core market timing edge, and then work on various aspects of the approach until we've got something worth trading. Back in early 2011, I presented an hour-long talk at the 2011 PowerUser Conference in Las Vegas that I'm roughly going to follow in this chapter. So this might not be totally new material to everyone, and in fact I know at least a couple people that have been trading this particular system (or a variation of it) live since then.

At PowerUser Conferences, a bunch of Wave59 users get together and listen to other Wave59 users present their trading tools and techniques. They're always a ton of fun, and I like them because I learn a lot too. The problem with conferences though is that the material tends to get lost in time once it has been presented, especially with some of the more complicated approaches. Because of that, I wanted to take this opportunity to talk about this particular system to make some of these techniques more widely accessible, especially since there are helper functions built right into Wave59 that make it easy to implement what we are about to do.

As I promised in the last chapter, we're going to be looking at a system based on Astrology. That actually might be a misnomer, as there is no actual Astrological lore involved, and everything is derived using computers and backtesting. The edge that we have is related to a new way of calculating planetary aspects, which is demonstrably better than the traditional method, at least from the perspective of financial markets. Before I can really get into the system itself, I'd like to take a few pages and describe the background for how all of this works, which will also require some introductory material for those readers who might not be familiar with Astrology in general.

First of all, don't get squeamish if you've never heard of using planets to time markets before. It's always best to be open about these things, and let the numbers speak for themselves and guide you to the right answers. I have no clue how planets influence markets, given they are so far away, but I can build mechanical systems using them, and I can use those systems to make money. If I can do that, then sure, I'm a fan of Astrology. If it didn't work, I sure wouldn't be wasting time on it. You can replace the word "Astrology" in this paragraph with any other technique as well, including dollar-based stops, as this is my general approach to markets.

I've written quite a few Astro primers in other books, so you are welcome to check those out for more background information. The key point to note here is that we are going to be using **Spherical Astrology**, which is different from what you'll find elsewhere.

In traditional astrology, all the planets are drawn in a circle, with either the Sun or the Earth at the center. The planets are then placed at the degree values they would appear at had the viewer been standing in the center.

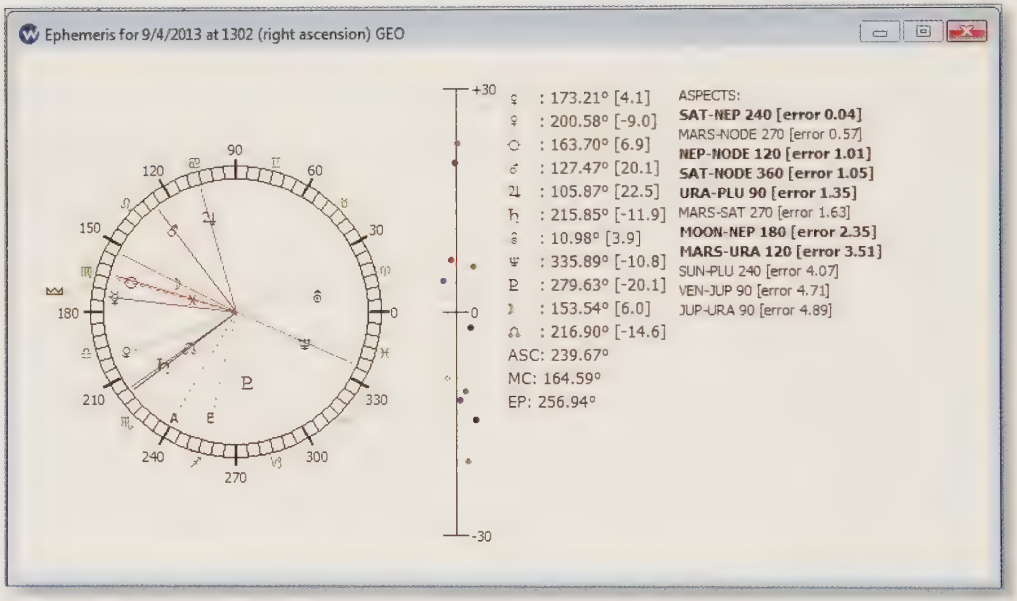


Figure 7.1: Ephemeris within Wave59

This is what Wave59's ephemeris looks like at the moment I'm writing this sentence. The round wheel at the left has all the planets, each with a different color and symbol. This is a geocentric, or Earth-centered, wheel, so the planets appear as they would as seen from Earth.

If you look at the far right, there's a list of various aspects, sorted by how exact they are. The top one is "Sat-Nep 240," meaning that Saturn and Neptune are 240 degrees apart from each other. This is important because 240 degrees is the same as one side of a triangle, had you drawn one inside that wheel.

You can see what this looks like visually in Figure 7.2:

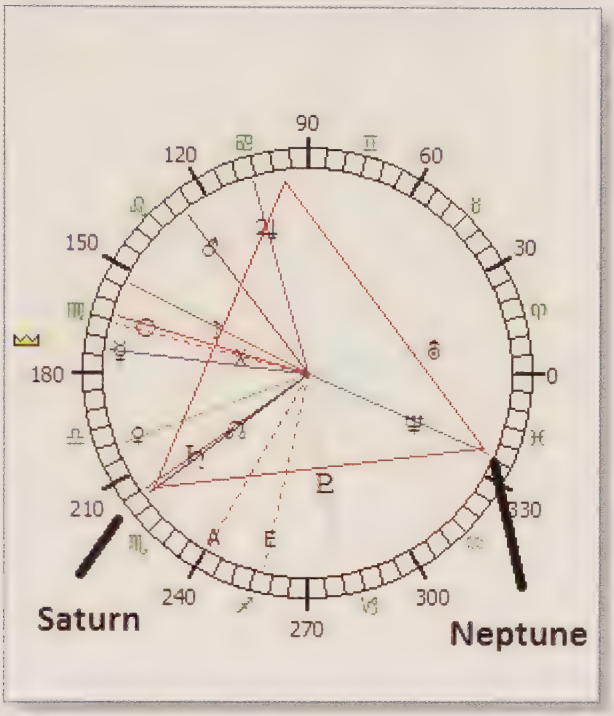


Figure 7.2: Trine Aspect Sat-Nep 240

I've zoomed in and labeled it to make it as clear as possible. Saturn's position is in the lower left-hand corner, and Neptune's is in the lower right. If you connect the dots between them, they set up a triangle shape. No planet is at the top of the triangle, but that's OK. This relationship is considered important to astrologers, as are all relationships that create shapes like this, such as squares (90 degrees), pentagons (72 degrees), hexagons (60 degrees), etc.

The big problem with these sorts of traditional aspects is that they just don't test very well on the stock market. Every now and then you can find something that sort of seems to work, but for the most part, you're not going to be able to build a good mechanical system using this information. There's a reason there aren't very many astro-based systems floating around – they don't work very well.

However, just because we can't build good astro systems doesn't mean that astro doesn't work. It only means that the traditional methods leave a lot to be desired, so we have to try and figure out new ways to use this information. It was really only when I started approaching this subject from a different angle that I made a lot of progress, and Spherical Astrology was a huge help.

If you look at these wheels, you'll realize that they are two-dimensional. We are looking at the solar system from the top-down, as if we were floating above it. If we were using Sun-based (heliocentric) Astrology, our chart would actually be a sort of miniature model of the Solar System. But we know that the physical world has more than just two dimensions. The planets are in orbit around the Sun, but that orbit isn't totally flat, with all the planets exactly in one plane. They actually move up and down.

If you take a look at the middle section of Wave59's ephemeris, you'll see a section that looks like:

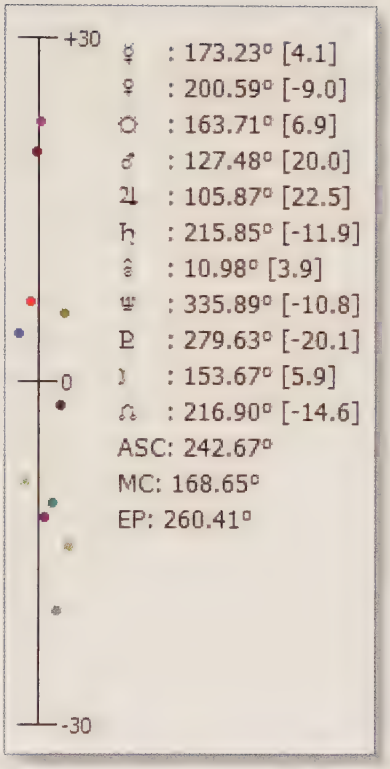


Figure 7.3: Declination of Planets within Ephemeris

There are two things here that are important. The first is the vertical graph on the left that runs from -30 to +30. This is the declination plot, and shows which planets are above 0 degrees and which are below it. Zero degrees is the Earth-Sun plane, which is our reference point. You can see that the planets are scattered about all over the place, with some being very high (like Jupiter in this example, at 22.5 degrees) and others being very low. The numbers to the right give you the actual numerical values for everything. For example, if you look at the top entry, there's a symbol for Mercury (looks like a little guy with horns on his head) which is at 173 degrees on our wheel, but also at 4.1 degrees above horizontal.

What this means is that traditional astrology is really simplifying things, and they've done so enough that the aspect values don't hold up very well in testing. What Spherical Astrology does is redefine the aspects, considering all three dimensions.

Let's go back to our idea of important aspects, and consider the square. A square aspect would be formed whenever two planets were exactly 90 degrees apart from each other. In our example for today, Mars and Saturn are 90 degrees apart, so they're making a square formation:

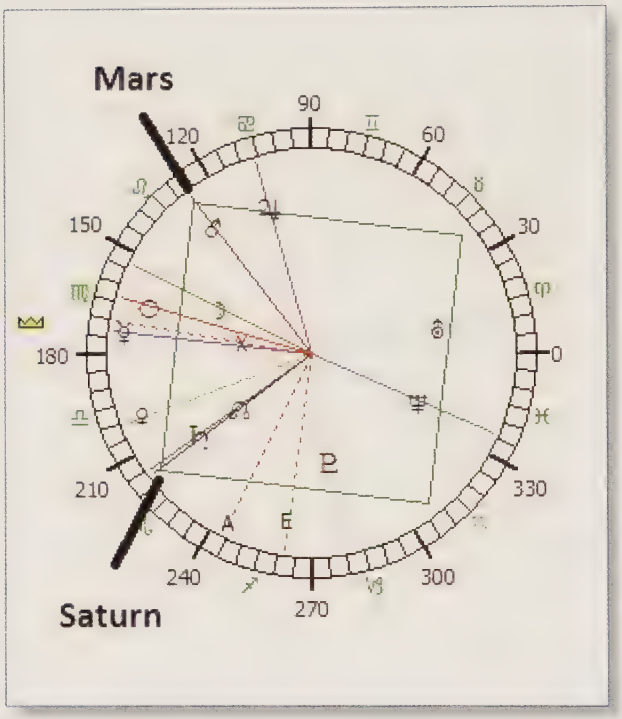


Figure 7.4: Square Mars-Saturn

However, remember that this is only as viewed from the top down. To really make a square, we have to consider the system in three dimensions. The easiest way to visualize this is to imagine a glass sphere, with the planets as dots moving around the outside of it. If we have a glass sphere, then imagine a square piece of cardboard, cut so that it exactly fits into the inside of this sphere, with all four corners touching. A true "square" aspect really only would occur when planets arranged themselves on the tips of our piece of cardboard.

In the case of Mars and Saturn in the chart above, that would only happen if both Mars and Saturn were exactly at zero degrees declination, both basically at the equator of the sphere, with the cardboard being horizontal to the ground. As soon as we start lifting

one corner of that cardboard up, the opposite corner will go down, and when viewed from above, two planets sitting on the corners will no longer appear to be 90 degrees apart from each other.

So what we've done is redefine what it takes to make an aspect, and we are thinking in terms of various shapes inside a sphere, rather than simple aspect differences in a two-dimensional plane. The famous astrologer and mathematician Johannes Kepler was obsessed with this sort of thing, so I wonder if perhaps traditional astrologers in the modern sense are using techniques that were dumbed down, as it's clear that Kepler, at least, had looked into this a long time ago.

Rather than computing aspects in terms of pure degree value differences, Spherical Astrology lets us compute a new measure, the ratio of the distance of two planets to the diameter of the sphere. Without a computer, this would be impossible, but Wave59 makes it easy, and provides functions within QScript to do so. Specifically, we can call the following function within our scripts:

```
astro_xyz_ratio(longitude1, longitude2, declination1, declination2)
```

This function will return the ratio between any two planets and the diameter of the sphere, given the longitude and declinations of the two planets. The value returned is between 0 and 1, but I like to multiply by 180, which gives me a ratio value that I can more easily compare to the old aspects with which I'm already familiar.

Once we've got our theory, and are able to program it, we can then apply it to markets and see how things work. In this case, when I came up with Spherical Astrology, I ran a ton of tests trying to see how various planetary pairs did at predicting the short term directional movement in various markets. I don't have those scripts anymore, but they're pretty easy to re-create.

In the appendix, you'll find a simple testing script (Earik_System_BookSphericalTester). It asks for two planetary pairs, which are keyed to numeric values so you can run them through the optimizer, then buys when the aspect ratio between those pairs crosses a threshold number. It then exits some fixed number of bars after the entry. The idea behind this script is not to build any systems, but more to poke around and see if we can find anything worth looking at further.

When applying this script to the Dow, you can find all sorts of interesting relationships. For example, Venus seems to be an important planet for markets. When the Venus-Sun ratio goes above 10, the indication is to buy and hold for 50 days, at which point you exit the position. That simple approach generated the following system report:

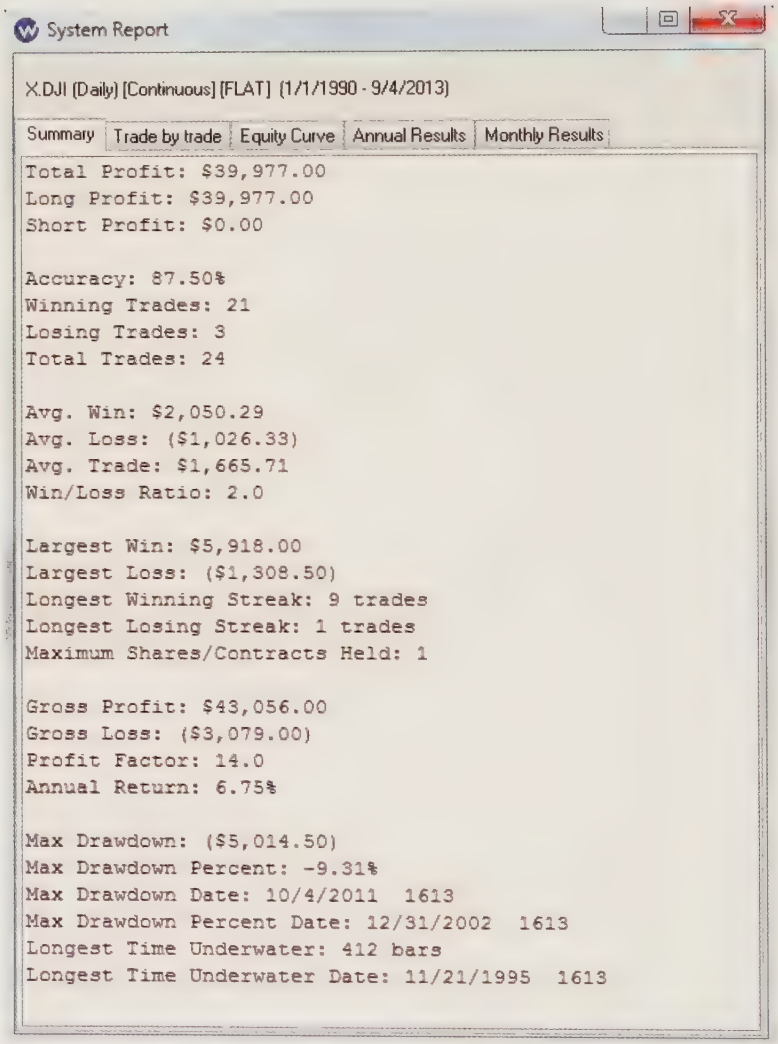


Figure 7.5: Simple Venus-Sun Aspect System

Check out the accuracy and win/loss ratios. 87% accuracy with a win/loss of 2.0? Wow! We've also got a very small drawdown, and a decent amount of profit generated. What does the equity curve look like?

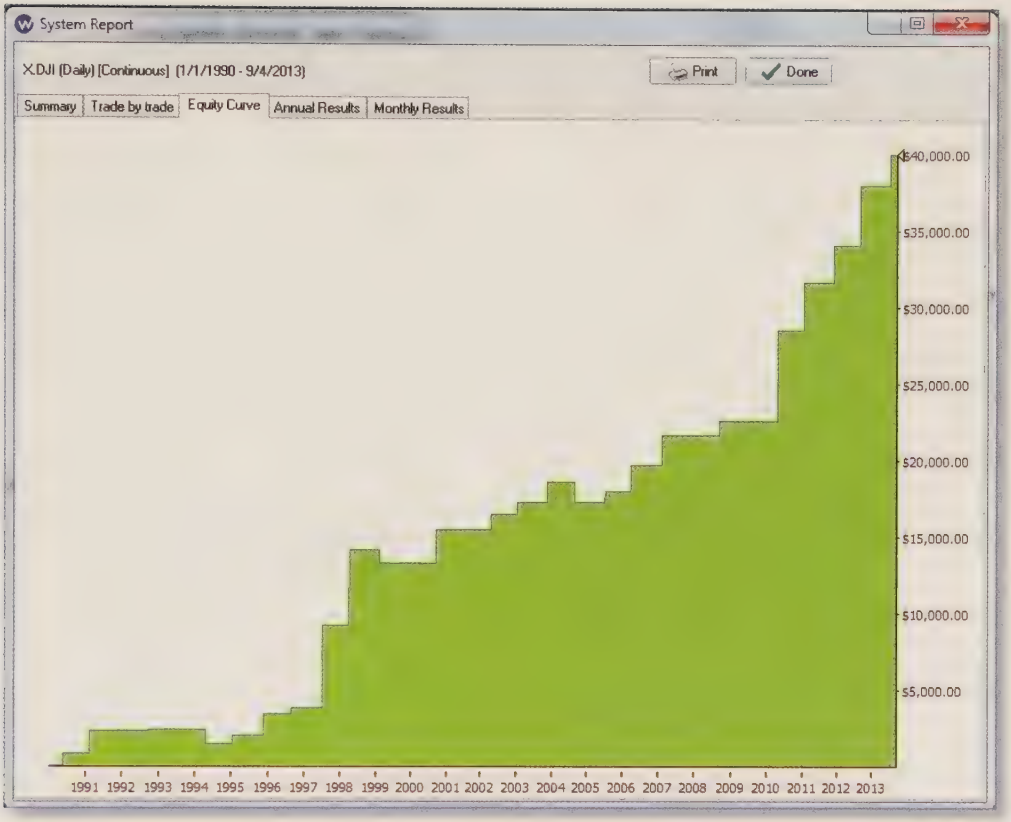


Figure 7.6: Sun-Venus Aspect System Equity Curve

That's a nice equity curve... I like it!

Ok, at this point, we need to sit down and ask ourselves what the weaknesses are to this strategy. I can think of two of them. First of all, and most importantly, there are only 24 trades. This is a little over 20 years worth of data, so we're getting into a William Tell type of situation here. One trade a year isn't going to cut it, at least not by itself. The second weakness is that we've only looked at the long side, and we're testing on the Dow, which mostly goes up. So maybe this pattern works, or maybe we're just getting lucky here. Don't confuse brains with a bull market, as they say.

Let's check the short side of this pattern and see what it has done. We'll do the same thing, sell when our ratio crosses under some threshold level, and hold.

Optimization Results

	Parameters	Profit/Loss	Accuracy	Num. Trades
1	#1#=10,#2#=10	$3,553.00	43.48%	23
2	#1#=10,#2#=30	$2,739.00	52.17%	23
3	#1#=10,#2#=20	$2,576.00	52.17%	23
4	#1#=60,#2#=10	$2,025.50	44.83%	29
5	#1#=70,#2#=30	$1,369.50	45.00%	20
6	#1#=80,#2#=30	$100.00	100.00%	1
7	#1#=80,#2#=10	($105.00)	0.00%	1
8	#1#=80,#2#=40	($174.50)	0.00%	1
9	#1#=80,#2#=20	($177.00)	0.00%	1
10	#1#=70,#2#=20	($758.00)	45.00%	20
11	#1#=80,#2#=50	($790.00)	0.00%	1
12	#1#=70,#2#=10	($1,671.50)	50.00%	20
13	#1#=60,#2#=20	($3,034.50)	44.83%	29
14	#1#=10,#2#=40	($4,921.50)	39.13%	23
15	#1#=20,#2#=10	($5,109.00)	37.93%	29
16	#1#=30,#2#=10	($6,066.50)	44.83%	29
17	#1#=70,#2#=40	($9,377.50)	40.00%	20
18	#1#=20,#2#=20	($10,262.50)	34.48%	29
19	#1#=60,#2#=40	($11,197.00)	31.03%	29
20	#1#=60,#2#=50	($12,276.00)	41.38%	29
21	#1#=10,#2#=50	($12,881.50)	30.43%	23
22	#1#=20,#2#=30	($13,652.50)	31.03%	29
23	#1#=70,#2#=50	($14,281.50)	35.00%	20

Figure 7.7: Sun-Venus Aspect Threshold Levels Optimization

Figure 7.7 shows the results from the optimization run for different threshold parameter values as well as holding periods. It displays all results, with the best ones at the top. There were a couple of solutions that managed to make a little money, but not a lot, and by far the majority of them were terrible losers. Venus-Sun can't go short the Dow, which means we're stuck with a long-only pattern that trades only about once per year. 24 instances isn't significant, so as good as this pattern looks, it's not something I'd want to trade.

In the same vein, we can eliminate all planetary combinations that just produce too few trades. If we can't generate enough cases to get numbers we can trust, then there's no

point in investigating patterns like this further. As we start going through these patterns, one planet in particular really starts to grab our attention: the Moon. The Moon is fast, and it makes one revolution around the wheel in a month, which gives us a ton of cases to analyze.

Ok, we know we want to focus on the Moon, so now let's figure out what other planets can be combined with the Moon to produce a tradable system. We need two of them, since we're using ratios. Again, a perfect task for our optimizer. Notice how I tend to use the System Optimizer more to help sift through data, rather than to try and solve for silver bullet parameters that make things work. A brute-force optimizer can be your best friend or your worst enemy. It all depends on how you use it.

Optimization Results

	Parameters	Profit/Loss	Accuracy	Num. Trades
1	#1#=2,#2#=170	$105,551.50	62.63%	289
2	#1#=2,#2#=160	$94,821.00	63.67%	289
3	#1#=2,#2#=165	$92,508.50	61.25%	289
4	#1#=2,#2#=155	$84,601.50	65.40%	289
5	#1#=1,#2#=165	$82,805.00	62.76%	290
6	#1#=9,#2#=80	$82,265.50	62.94%	313
7	#1#=2,#2#=175	$82,145.00	62.98%	289
8	#1#=1,#2#=160	$81,526.00	63.45%	290
9	#1#=3,#2#=175	$80,827.50	61.72%	290
10	#1#=2,#2#=145	$79,949.00	66.21%	290
11	#1#=2,#2#=140	$79,864.00	64.48%	290
12	#1#=9,#2#=35	$79,433.00	60.53%	304
13	#1#=4,#2#=175	$79,103.50	59.73%	298
14	#1#=2,#2#=135	$77,265.50	64.83%	290
15	#1#=9,#2#=85	$76,961.50	62.94%	313
16	#1#=2,#2#=150	$76,826.50	65.74%	289
17	#1#=6,#2#=175	$75,868.00	61.29%	310
18	#1#=9,#2#=30	$75,692.50	60.70%	285
19	#1#=1,#2#=155	$75,294.00	63.10%	290
20	#1#=5,#2#=150	$74,663.50	64.10%	312
21	#1#=1,#2#=170	$74,433.00	62.07%	290

Figure 7.8: Lunar Ratio System - 15 day Hold Optimization

Figure 7.8 shows the top solutions for buying a Lunar ratio and holding for 15 days. I'm seeing Venus at the top (#1# = 2), along with Mercury, Sun, and a lot of other planets. We probably could go with any of them. Notice the number of trades at around 300. I am much more comfortable with this sort of pattern than the last one with only 24 trades.

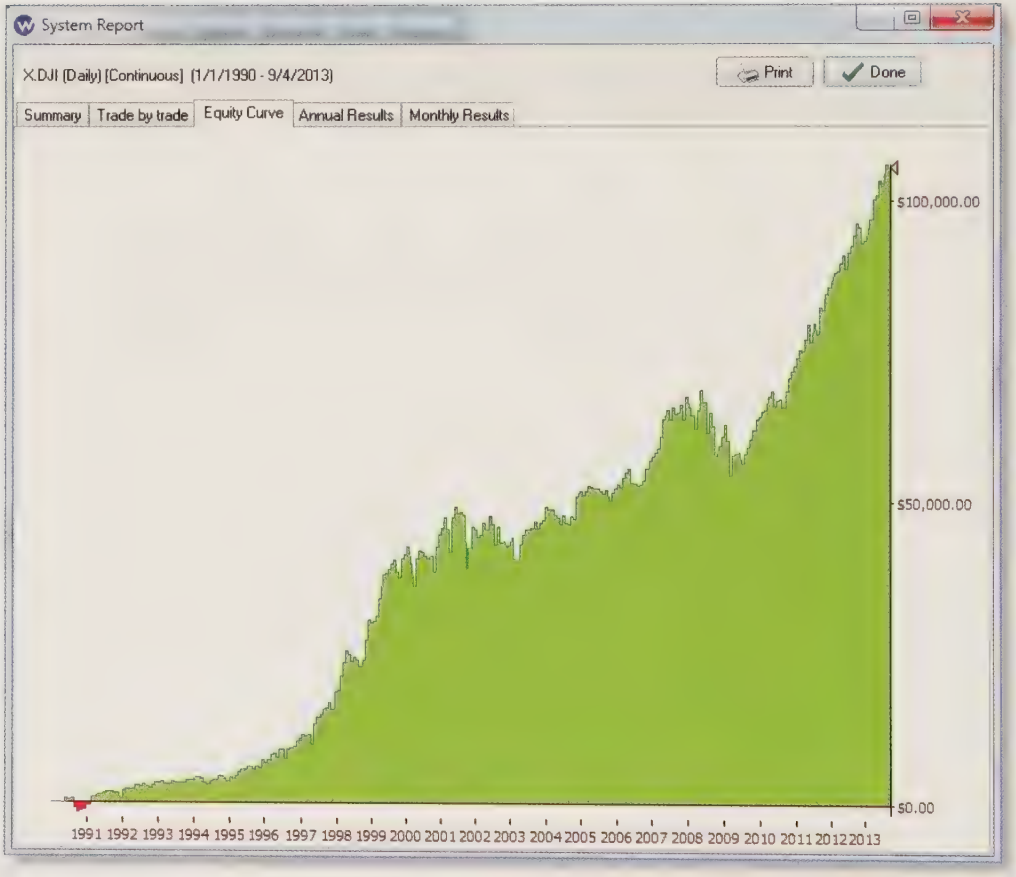


Figure 7.9: Moon-Venus Ratio System Equity Curve

Figure 7.9 shows the equity curve from the best solution, a Moon-Venus pattern. I could live with that.

At this point, we've got to make a choice about which direction to go with our second planet, and Venus is looking pretty good, at least by the numbers. However, back when I was building this system, I ended up selecting the Sun instead. I like my systems to be

based on something that I can intuitively grasp, and when presented with two directions, I will take the direction that has the stronger theoretical argument behind it. The Sun and the Moon are powerful forces in our world, and are involved in tides and seasons, among a whole slew of other things. It's not a huge leap to think that they'd be involved in markets as well. I'm not able to make that same statement when it comes to Venus though, so when it came down to Moon-Sun or Moon-Venus, I made an executive decision and went with Moon-Sun. (That's not to say that there isn't an awesome system here for the Venus fans who are willing to dig for it.)

At this stage, we've got the basis behind our system, and it's the Moon-Sun relationship. Now we have to really try to understand what is going on with this pattern, and for me that usually means plotting lots of charts and studying them by hand. I like seeing the data in some visual way, and when I can look at it in chart format, I've got a better chance of understanding what I'm working with. This particular step will vary depending on what specifically you are building, but in my case there's almost always some chart reading in the early stages at some point.

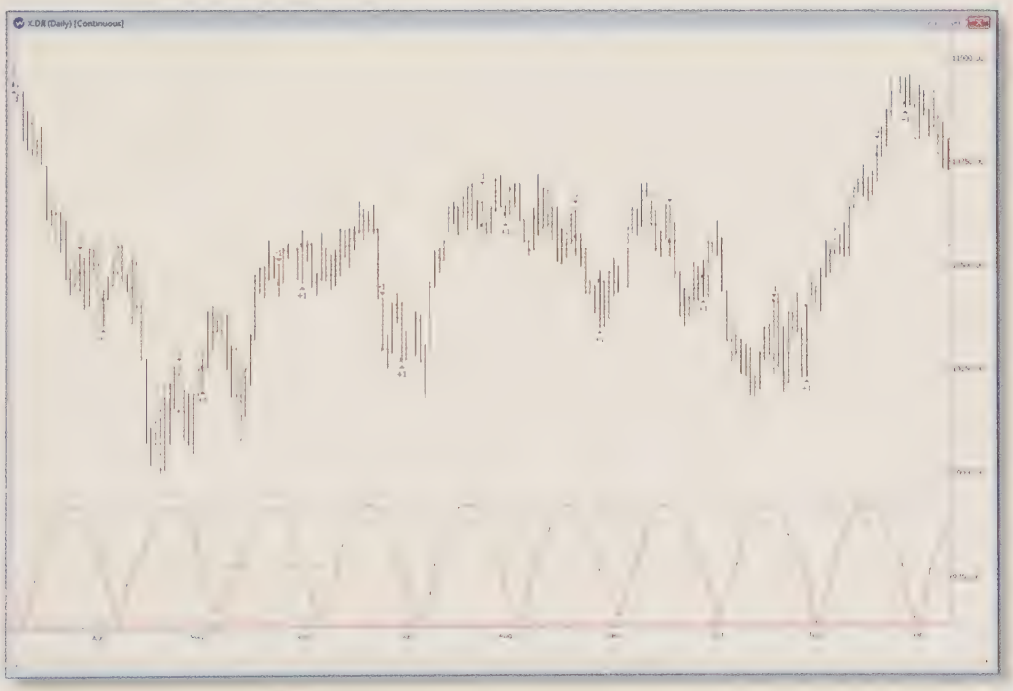


Figure 7.10: Plotting the Ratio as an Indicator

In the case of this method, we can plot the ratio as an indicator, and study it to understand what is going on. You can see an example of this on the chart in Figure 7.10 on the previous page. After looking through the chart, and running more tests, I eventually came to the conclusion that the market wants to rally on Quarter Moons, and sell off on New and Full Moons. Remember we're talking about Spherical Quarter Moons here, which are a little different from what you'll see in the Farmer's Almanac. The green line in the middle of the graph is the "buy" line, and the two red lines are the "sell" lines. The oscillator crowd has this one covered.

You can see how our signal generator is starting to take shape. We have our basic idea, which is that Spherical Moon phases have an impact on the market, and we have enough data to show that following along with these signals results in profits. The next step is to determine exactly where we should put the buy/sell trigger lines. I've got them in their theoretically obvious places in Figure 7.10, but we need to double check a range of solutions before we fix those levels. Back to the Optimizer!

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=70	$15,539.00	50.91%
2	#1#=72	$19,247.00	50.76%
3	#1#=74	$20,548.00	51.37%
4	#1#=76	$21,365.00	51.83%
5	#1#=78	$26,284.00	51.37%
6	#1#=80	$30,496.00	51.52%
7	#1#=82	$34,312.00	52.13%
8	#1#=84	$31,944.00	51.52%
9	#1#=86	$39,508.00	52.13%
10	#1#=88	$48,901.00	52.74%
11	#1#=90	$45,313.00	52.74%
12	#1#=92	$51,630.00	52.74%
13	#1#=94	$67,532.00	53.35%
14	#1#=96	$64,603.00	53.35%
15	#1#=98	$68,021.00	53.51%
16	#1#=100	$66,082.00	53.05%
17	#1#=102	$79,052.00	53.51%
18	#1#=104	$78,710.00	53.51%
19	#1#=106	$76,574.00	53.81%
20	#1#=108	$87,671.00	54.57%
21	#1#=110	$86,419.00	54.73%
22	#1#=112	$84,599.00	54.88%
23	#1#=114	$85,105.00	55.64%
24	#1#=116	$72,912.00	54.88%
25	#1#=118	$98,447.00	55.95%
26	#1#=120	$89,148.00	55.49%

Figure 7.11: Lunar System Buy Line Optimization

Figure 7.11 lists the results of moving the buy line to different levels. The theoretical place for it should be at 90, solution number 11, but you can see that the higher values are much more profitable, and the upward trend to these numbers really shows that we want to use a larger value here rather than a smaller one. Back when I first built the system, I used a value of 112. Currently, it looks like 118 has been more profitable, and I can't say that I'm surprised to see a parameter move around. What is more important is that all of these parameters work, and as long as we stay slightly ahead of the exact

middle, we should remain close to optimal. I'm going to keep using 112 here because I'd like to keep the results from early 2011 until today in the out-of-sample category.

Now we have to take a look at the sell numbers. Note that we could also have done all the optimization in one step, but three variables at once results in quite a few combinations, so sometimes it's easier to do it in small chunks, then fine tune a bit at the end. It's not like we're going to be able to get the exact optimal parameter anyway, since they will change as we move forward.

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=3,#2#=169	$104,262.00	56.62%
2	#1#=3,#2#=171	$102,482.00	56.79%
3	#1#=3,#2#=173	$103,522.00	56.62%
4	#1#=3,#2#=175	$90,920.00	55.79%
5	#1#=3,#2#=177	$74,853.00	53.34%
6	#1#=5,#2#=169	$97,941.00	55.49%
7	#1#=5,#2#=171	$96,161.00	55.49%
8	#1#=5,#2#=173	$97,201.00	55.49%
9	#1#=5,#2#=175	$84,599.00	54.88%
10	#1#=5,#2#=177	$68,532.00	52.77%
11	#1#=7,#2#=169	$98,505.00	53.69%
12	#1#=7,#2#=171	$96,725.00	53.69%
13	#1#=7,#2#=173	$97,765.00	53.56%
14	#1#=7,#2#=175	$85,163.00	53.03%
15	#1#=7,#2#=177	$69,096.00	51.33%
16	#1#=9,#2#=169	$77,379.00	52.71%
17	#1#=9,#2#=171	$75,599.00	52.60%
18	#1#=9,#2#=173	$76,639.00	52.37%
19	#1#=9,#2#=175	$64,037.00	52.03%
20	#1#=9,#2#=177	$47,970.00	50.80%
21	#1#=11,#2#=169	$92,723.00	53.07%
22	#1#=11,#2#=171	$90,943.00	52.97%
23	#1#=11,#2#=173	$91,983.00	52.66%
24	#1#=11,#2#=175	$79,381.00	52.15%
25	#1#=11,#2#=177	$63,314.00	51.03%
26	#1#=13,#2#=169	$101,453.00	53.26%
27	#1#=13,#2#=171	$99,673.00	53.16%
28	#1#=13,#2#=173	$100,713.00	52.87%
29	#1#=13,#2#=175	$88,111.00	52.37%
30	#1#=13,#2#=177	$72,044.00	51.49%

Figure 7.12: Lunar System Sell Lines Optimization

A number of combinations of the sell thresholds need to be tested, and we find that all combinations produce profitable results. That's the most important point to test. We want to make sure we're not basing our trading decisions on a silver bullet parameter

set, and we also need to be able to expect our system to perform when it moves onto a new optimal parameter set in the future. Originally, I had decided on a 5/175 threshold combination with this method, which you can see as solution 9 in the list. It is actually closer to the bottom of the results here, but I believe it's still a good choice for theoretical reasons. The New Moon/Full Moon numbers are 0 and 180, but the indicator won't always show those because they might happen at times other than the close, and also because Spherical New Moons and Full Moons are more like eclipses, and won't always happen exactly each month. So I figured a pad of 5 degrees each side would be a good cushion, and ended up at 5/175. Again, sometimes the optimization is just there to show that the numbers around your solutions are all safe. Trying too hard to get the "best" solution among those numbers is usually counter-productive, as you are almost guaranteed that the optimal solution looking forward is going to be different than the optimal solution looking back.

As a quick tangent, I'm writing this book in September of 2013, and you might have noticed that in doing optimization runs, I'm using the entire data set. If you've read other system-related books, you've probably heard developers talk about splitting a data set up into different chunks to help avoid curve-fitting. This is a very common practice, and what you do is hold a few years out of your data set at the end (and sometimes also at the beginning). You then go ahead and perform all your development work on the truncated data set, and then when you are done, you run your system forward on the data you held out. The idea is that if a system is good, it needs to be able to run on the unseen data set as well, and the unseen set is used as a sort of final check against over-optimization.

First off, let me just say that I usually do keep a year or two out-of-sample, because I do like to see how things run on unseen data that I haven't had a chance to mess with. However, the reality is that this unseen data isn't really unseen, and I have come to question whether or not this really adds anything to the development process. The reason I say this is because there is no system that will make it through to live trading unless it passes the test of the unseen data. Therefore, the unseen data actually does become part of the development set, because the only systems that get put into production are ones that the developer has seen run on this out-of-sample data. So what happens is that systems do get optimized over the entire period, even including the out-of-sample data, it's just that the optimization has been split into two segments,

one segment for "official" development, and one that happens afterwards. But curve-fitting is curve-fitting, and doing it incrementally isn't going to make you any safer than doing it all in one pass.

Here's a good analogy. Let's say a developer uses a technique based completely on random numbers to generate stock picks. Specifically, all trading decisions are generated by a room full of well trained, dart-wielding monkeys. Put a page of the Wall Street journal up on the wall, then let the monkeys loose, and see what stocks to buy by looking at where the darts stick. The developer pokes and prods the monkeys, and figures out some way to make them get as many lucky throws in there as possible. When the monkeys are in top shape, he then tests them out using unseen data to make sure everything is in working order. In some tests, the monkeys' performance suffers, and then he has to go buy different monkey chow, and change the plants in the terrarium, and adjust whatever parameters he is trying to tweak that affect his monkeys' stock picking ability. He then goes through another round of tests like this, until finally the monkeys are able to ace the unseen data trials. So now, after passing the out-of-sample data tests successfully, are the monkeys ready to trade with live money?

If you play with the numbers enough, eventually you'll find things that appear to work, even though they don't. And if you do this enough times, and find enough systems, then some of them will make it through the out-of-sample runs, and some won't. The fact is, if you've got monkeys throwing the darts, you're going to lose money in live trading no matter how many times those monkeys managed to pass your out-of-sample tests.

Coming back to our system, we now have our base signal line, as well as the trigger levels where we are going to be buying and selling. Let's put it all together into a full-blown reversal system.

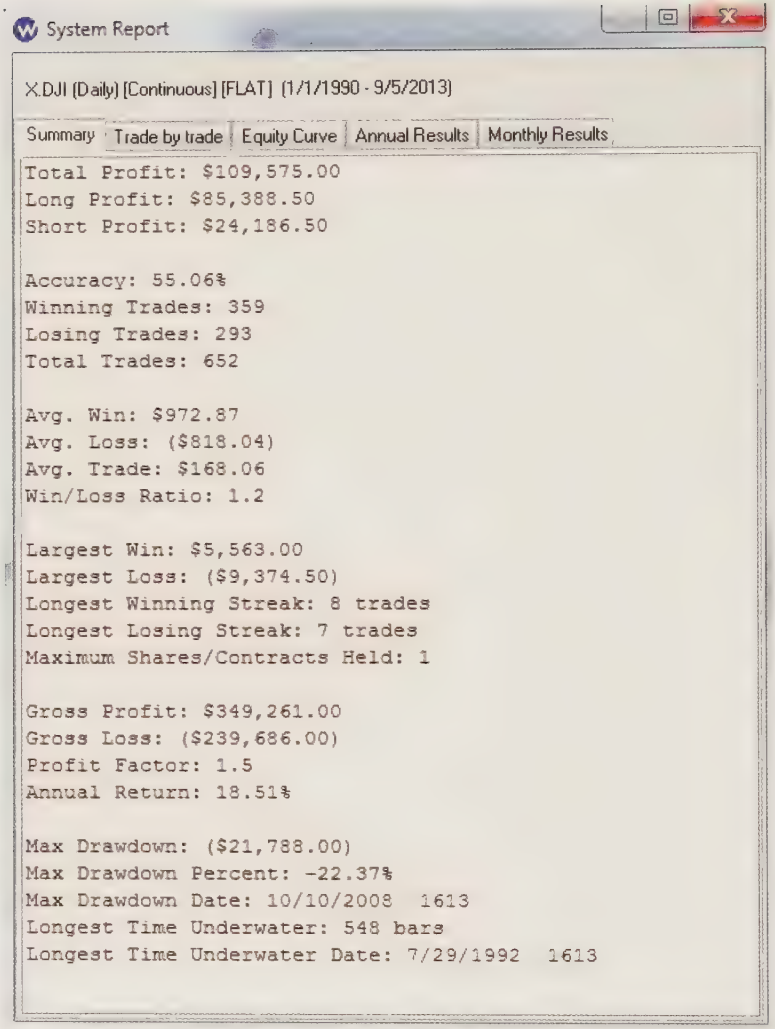


Figure 7.13: Lunar System with Optimal Signal Lines

These numbers are good, but not great. You can see the typical characteristics of a mean reversion/reversal system here. We've got 55% accuracy and a win/loss ratio of 1.2. That means we make our money both by being right more often than not, as well as with slightly larger wins than losses. That's a good position to be in, as it covers both the need for accuracy, as well as avoiding the issues that usually come with higher accuracy. Our average trade is $168, which is doable, but our drawdown slightly large.

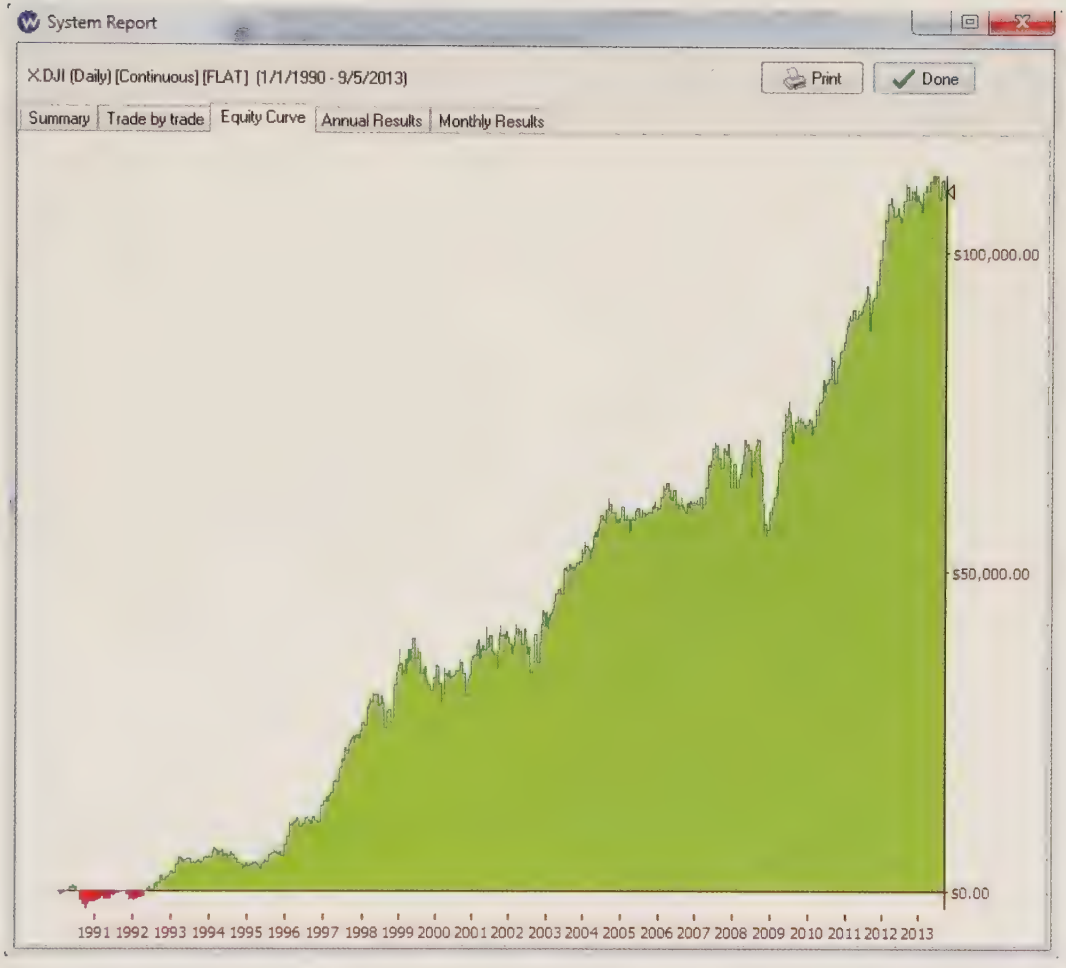


Figure 7.14: Lunar System with Optimal Signal Lines Equity Curve

The equity curve is good, showing a generally rising trend. It does take its lumps though, and 2009 in particular looked a little nasty. What you see here is a system that shows a lot of promise. When you reach a point like this, you have to try to figure out if there's a way you can help your system do its job a little better. There are usually clues in the system report, so going back over those numbers can often point you in the right direction. In our case, one of the things that really stands out to me is the discrepancy between Long Profit and Short Profit.

One of the things that most system reports do is break the profit down based on whether it was generated on the long side of the market or the short side. Right now,

almost 80% of our profit is coming from our buy signals, which is really lopsided. What I usually like to see is a balance between those numbers, but that can be difficult to achieve when it comes to trading stock indexes.

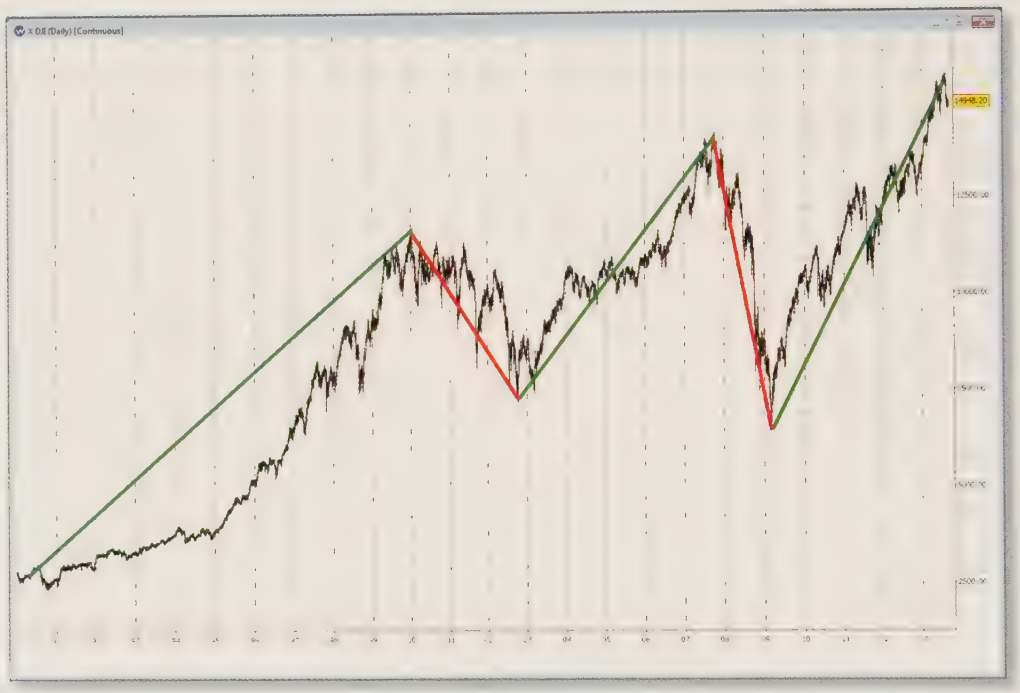


Figure 7.15: Dow Jones Big Trends

The data set we are working with is shown in Figure 7.15. I've sketched the big trends in with green lines for upward motion, and red lines for downward motion. You can see that usually the market is moving up. If you count the years that contribute to the green lines, you'll see that there are 20 of them. If you count the years that contribute to the red lines, there are only 4. This shouldn't be a surprise, as this is how stock markets move – they spend a lot of time going up, and then when they go down, they do it fast and hard, but it only lasts for a short period of time. That's why the average investor is usually making money in the stock market, except for the few times when they are totally getting slaughtered.

So we do want to go short, and that's important, but we also have to realize that it's harder to time those shorts, as they don't last very long. Because of that, I decided to

look and see if there was some filter I could apply to the short side of this system to help it perform a little better. What I came up with was a very simple rule:

> *Short Filter:*
> Go short only if the close is greater than yesterday's close

What I noticed after looking through a lot of the short trades on the chart is that a lot of times, the system got the signal right, but it got in a little too late. So the Lunar trigger really did call a drop correctly, but our rules just didn't get us in quickly enough. So the market dropped, we got in halfway down, and then the market bounced and we ended up taking a losing trade despite having the right idea. By only taking trades on an up-close, we avoid selling into the drop, and only go short when we're sure to get a halfway decent price. For the situations where we missed the signal, the system just ignores the short signal and stays long, basically merging two buy signals together. The results of this addition were quite dramatic:

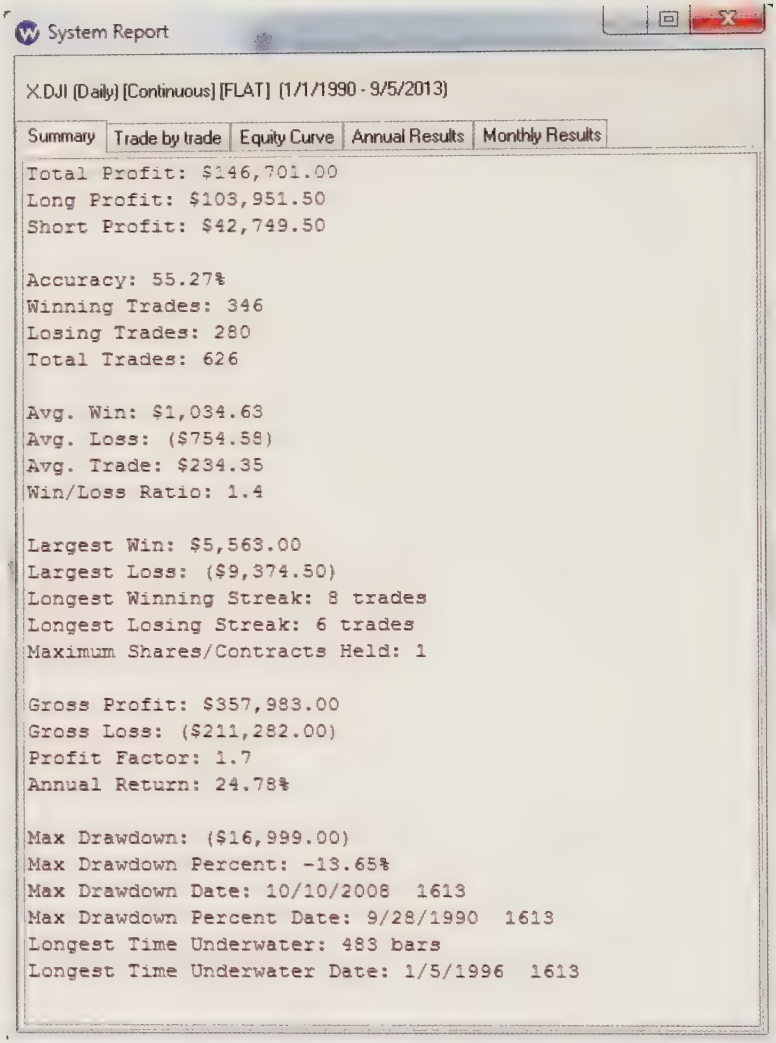


Figure 7.16: Lunar System with Short Filter System Report

Our total profit jumped by over 30% when applying this simple rule, and now we're up at $146k over this period. Our short profits doubled, moving up to $42k, and our long profits increased as well. The money made on the long side is still much larger than the money made on the short side, but I don't think there's a way around that given the market we are looking at. Our average trade got bumped up to $234, which is starting to feel pretty good, and our drawdown number dropped to -$17k, a 20% reduction. One simple tweak completely energized this system.

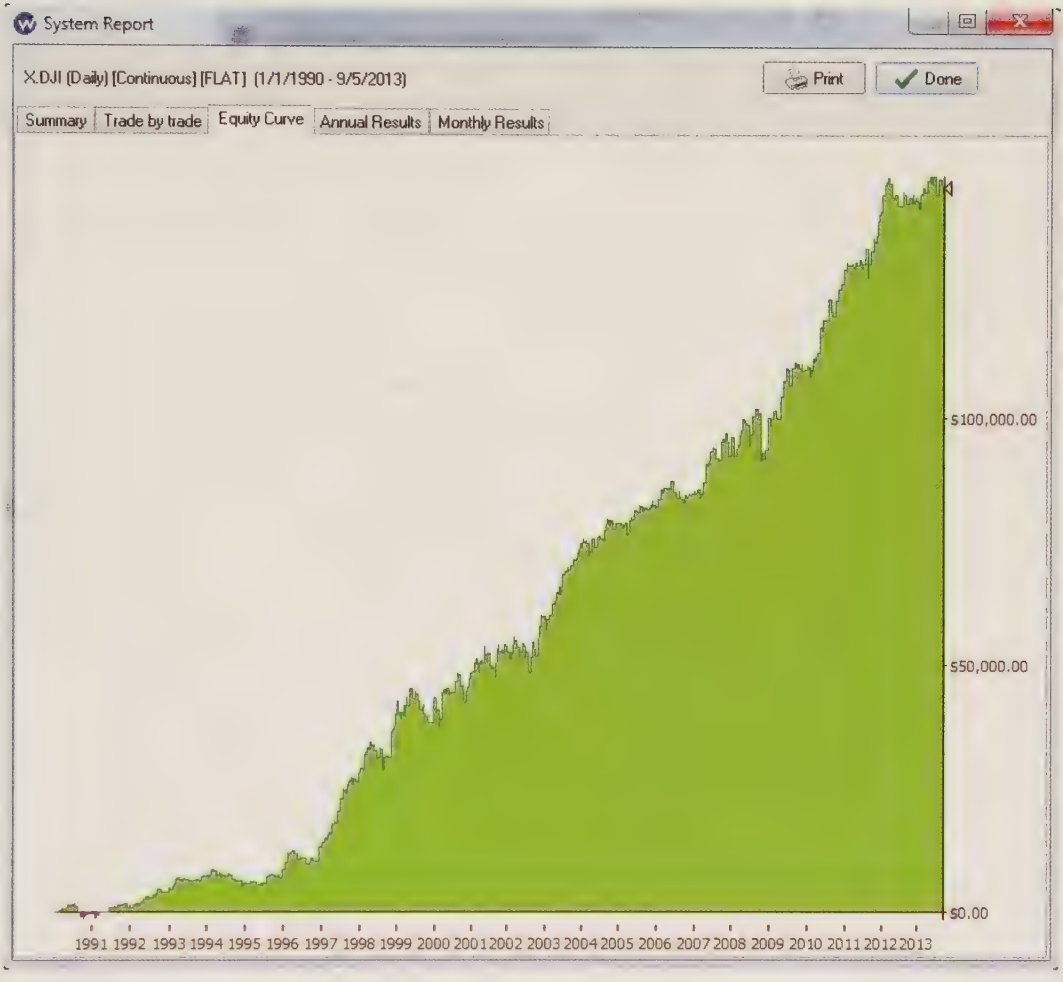


Figure 7.17: Lunar System with Short Filter Equity Curve

The positive changes are reflected in the equity curve as well. You can see how the drawdowns in 1999 and 2009 aren't nearly as nasty as they were before the filter was added. That tells you that the money being lost during these periods was actually happening on the short side, which might not be what we initially would have expected, given the fact that those dates corresponded to big market tops. The system has done especially well the last two years, which have been actual live trading years for it, since I first described this method with these parameters back in early 2011.

At this point, we've got a good system. It is a reversal system, and that does result in the approach having certain characteristics. Usually, that means around 50% accuracy, around 1.0 win/loss, with a slightly high drawdown value. None of that is bad, and I've traded plenty of systems that worked this way, but it's always a good idea to spend a little time and see if the system can be transformed into some other type of approach to see if things look even better. We've already looked at William Tell, which used quick profit taking to generate high accuracy and a very smooth (usually) equity curve. Let's try the same thing here with the Lunar system.

Rather than exiting at the first profitable exit, I've modified that approach to exit at some profitable close. Rather than the first profitable close, we can try exiting at the second profitable close, or the third. The general idea is the same, but trading at the close seems to produce larger gains in general, which is a good enough reason to do it that way. The added side benefit is you can sleep in too!

The modification we'll be looking at is to add an input that tells us at what profitable close to exit the trade. Everything else works the same way, and if we haven't had that many profitable closes by the time the next signal rolls around, we just go with that next signal, which results in a trade that is exactly the same as the original implementation. If we do get enough profitable closes in our favor, we take our profits, go flat, and wait for the next signal instead. Let's put this on the optimizer, and test this exit strategy.

Optimization Results

	Parameters	Profit/Loss	Accuracy	Num. Trades	Avg. Win	Avg. Loss	Avg. Trade	Win/Loss Ratio	Profit Factor	Max. Drawdown
1	#1#=1	$92,729.00	80.55%	874	$331.95	($829.21)	$106.10	0.4	1.7	($10,365.00)
2	#1#=2	$104,041.00	71.06%	857	$510.39	($833.81)	$121.40	0.6	1.5	($11,218.00)
3	#1#=3	$118,292.50	67.30%	847	$601.51	($810.72)	$139.66	0.7	1.5	($14,936.00)
4	#1#=4	$121,997.50	64.71%	833	$671.05	($815.30)	$146.46	0.8	1.5	($14,936.00)
5	#1#=5	$125,488.50	62.00%	829	$723.26	($781.80)	$151.37	0.9	1.5	($15,424.50)
6	#1#=6	$118,034.50	61.00%	818	$747.56	($799.37)	$144.30	0.9	1.5	($17,008.00)
7	#1#=7	$128,289.00	60.20%	804	$788.80	($792.16)	$159.56	1.0	1.5	($15,334.50)
8	#1#=8	$124,640.00	58.80%	784	$817.13	($780.36)	$158.98	1.0	1.5	($16,342.50)
9	#1#=9	$116,599.50	57.48%	769	$832.73	($769.02)	$151.62	1.1	1.5	($17,788.50)
10	#1#=10	$128,017.50	57.73%	757	$852.59	($764.26)	$169.11	1.1	1.5	($14,936.00)

Figure 7.18: Lunar System Nth-Profitable Close Exit Optimization

The fastest exit I tested was to exit at the first profitable close, and the slowest was to exit at the 10th profitable close. As expected, the faster exits are more accurate than the slower ones, but the slower ones make more money. Eventually, with a long enough count, the system will turn into the original reversal method, because we won't be able to have enough profitable closes before a new signal comes long. One important column

to pay attention to in Figure 7.18 is the win/loss column. Notice that it's 0.4 with a 1-close exit, which is a little small. Better than William Tell, but that is going to result in some nasty holes to dig out of eventually. The 2 and 3-close exits are better though, with 0.6 and 0.7 win/loss ratios, respectively.

Let's take a more in-depth look at exiting on the 2nd profitable close.

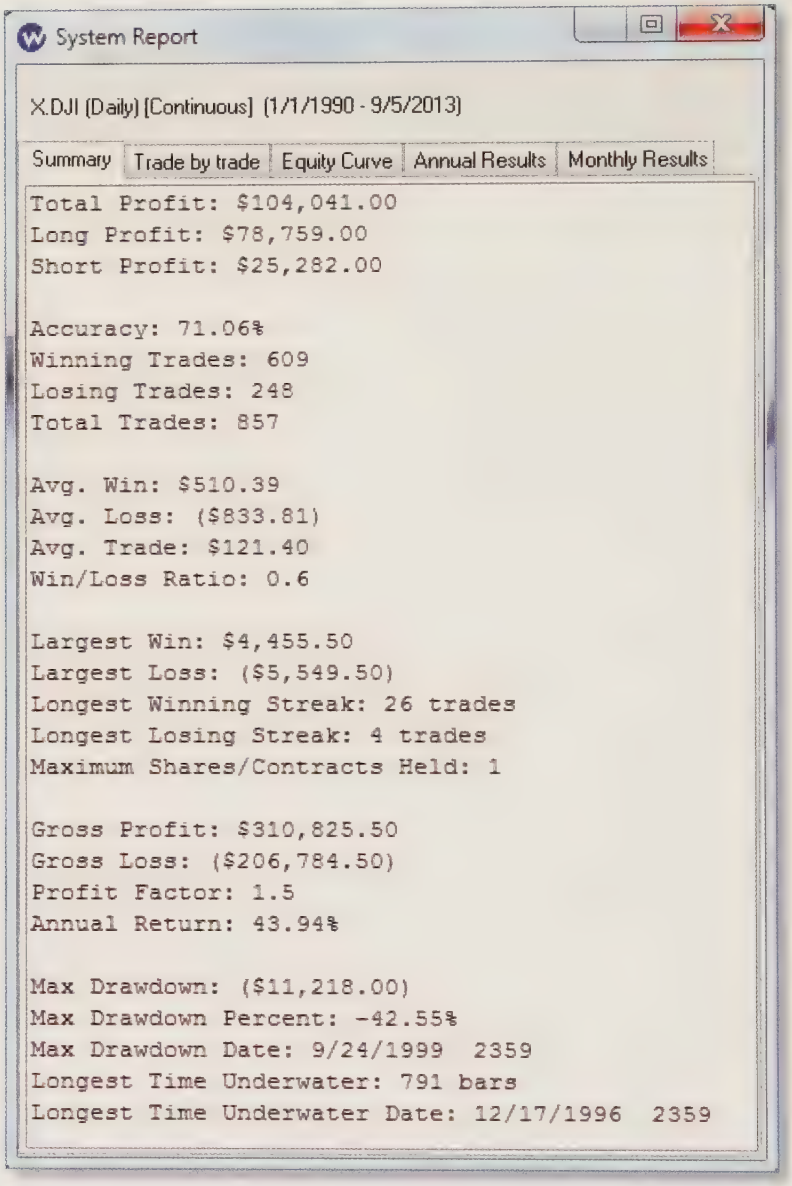


Figure 7.19: Lunar System 2nd Profitable Close System Report

When compared to our original system, we see that we've lost about $40k in profit off the top. Our accuracy is quite high, as expected, at 71%, with a win/loss of 0.6. Average trade has dropped, and drawdown has really dropped as well.

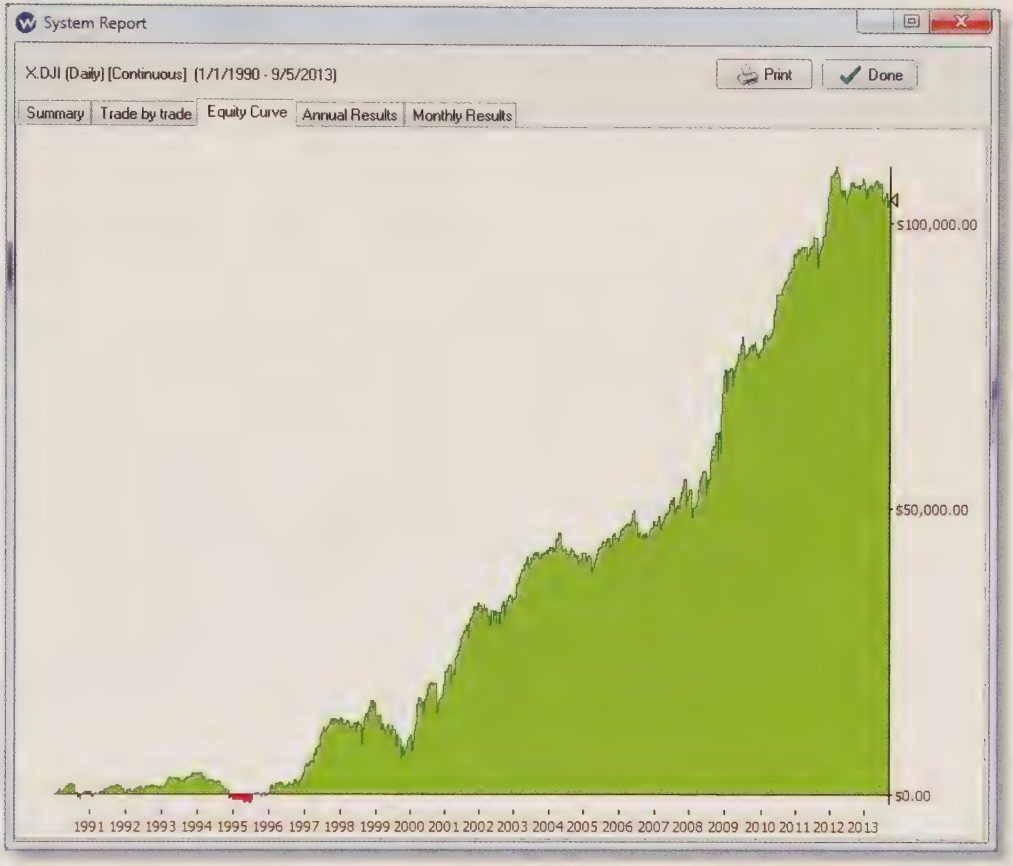


Figure 7.20: Lunar System 2nd Profitable Close Equity Curve

The equity curve is very similar, but slightly more sensitive. You can see that the original reversal system stuck to the 45 degree line a bit more closely than the fast exit version, although both of these are still great equity curves. How do we decide which is better?

When judging two very similar methods, it can be difficult to decide which method is superior. The first thing to consider is the equity curves. If one has a clearly better equity curve, that's the method I usually end up choosing as preferable. In this case, that's the original, but it's not a clear winner. The other thing I like to do is compare the profits generated compared to the drawdown value. If you divide those two, you get a metric that represents dollars earned per unit of risk, which is really what we're trying to optimize when trading. When looking at these two approaches, we can compute that metric as follows:

System	Total Profit	Max Drawdown	Profit/Drawdown
Original Reversal	$146,701	$16,999	8.62
2nd Profit. Close	$101,041	$11,218	9.00

Table 7.1: Original Lunar vs. 2nd Profitable Close Dollars Earned per Unit of Risk

In the case of dollars gained per dollars risked, the original system generated $8.62 per every dollar drawn down, and the 2nd profitable close system improved on that number to bring the ratio up to 9. So from the perspective of risk, the profitable exit method is actually better. You might make more money trading the first one, but if you end up using position sizing and compounding profits in a large account, you'll go faster with the second system. More on that in a later chapter!

After I had done the original optimization run, I liked most of the Nth-profitable close exits. I liked the 1st profitable close exit because it was very accurate, and I liked the other ones because they improved on the win/loss ratio of the first. So as a variation, I tried trading a bunch of them together as one unit. What you would do in this case would be to enter using multiple contracts, let's say 3. You would then take the first contract off at the 1st profitable exit, take the second off at the 2nd profitable exit, and then take the last contract off at the 3rd profitable exit. So you end up trading a blend of these methods together.

This is one of the ways I avoid trading systems that are overspecialized. Rather than solving for one optimal parameter set and trading that, I like to find systems that work for many different parameters, and then trade all the systems together. That allows me to trade the average result over a lot of parameters, and does a really good job of insulating the system from parameter migration over time.

This approach is sort of like hedging our bets. The odds of us being able to select the optimal parameter set out of a pool of many parameters are very low, but the odds go up the more guesses we get. So if we can trade three or four, or even ten, parameter sets all at once, we might end up hitting on the optimal solution and have a monster year. Of course, this is only going to work if the method itself is robust and generates profits across most of its parameter space, which is the key ingredient that you should aim for when developing systems. Then, if you've got that, you simply need to find a way to merge a bunch of solutions together into one system. This can include having the systems all vote on a single trade, or it can be actually trading a number of them together with multiple contracts. The Law of Averages can be your friend when it comes

to system testing, as the average results from many parameters traded in a good system will always be good. One parameter set might perform poorly, but the average itself will be very stable, and this is an extremely easy way to add a lot of reliability to your methods.

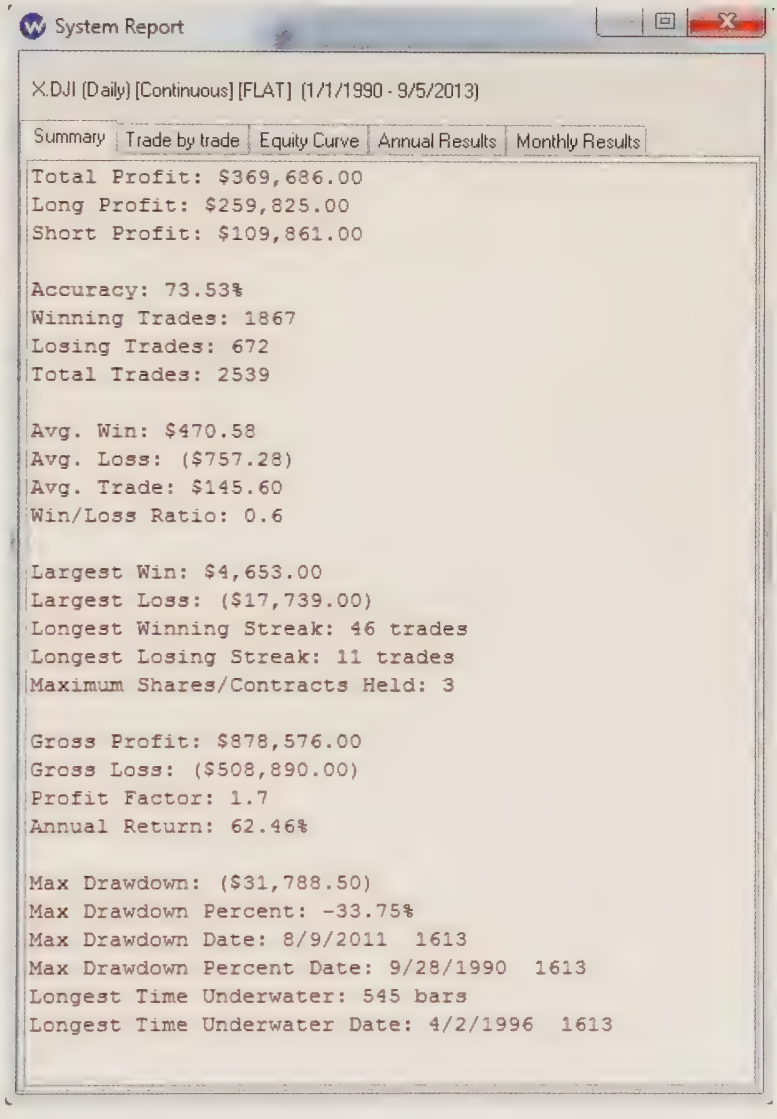


Figure 7.21: Lunar System Combined Exit Results System Test

Figure 7.21 shows the report from trading three contracts together, and exiting each one at its own profitable close. Since we're trading more contracts, our profit and drawdown numbers will be much larger than any of the single contract reports. You can see we've got great accuracy as well as a decent win/loss ratio. This exit style isn't going to give you a huge win/loss ratio, so you want to try to get it as close to 1.0 as possible.

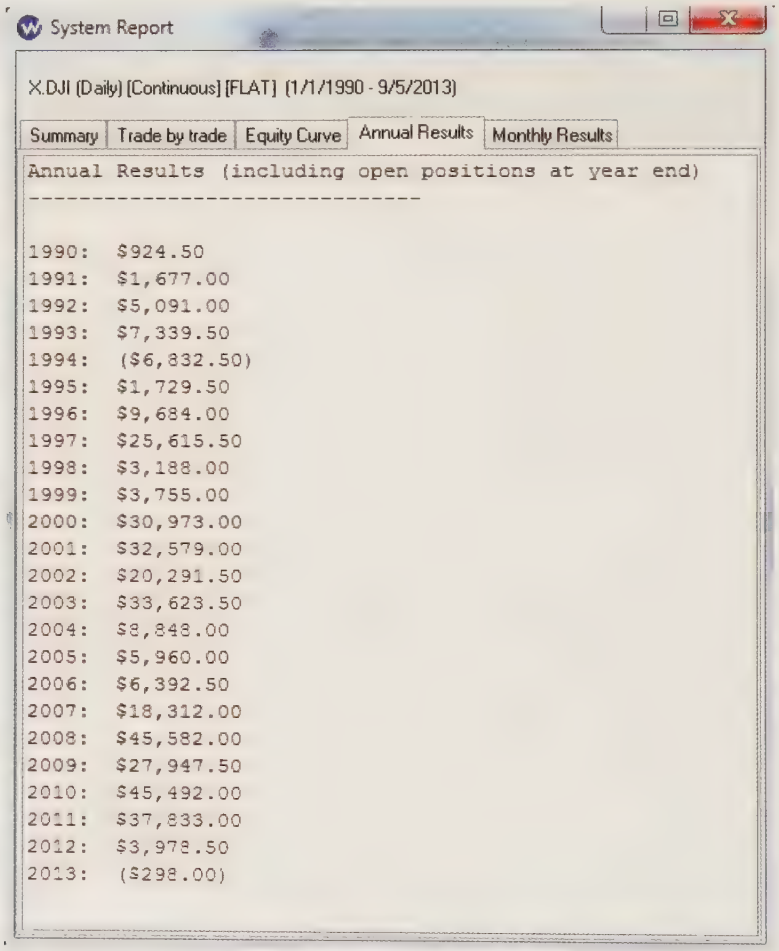


Figure 7.22: Lunar System Combined Exit Annual Results

Figure 7.22 shows the annual results page, which is sort of a numeric way of looking at the equity curve. This approach only had two losing years since 1990, and there's still enough time left in 2013 to get that one into the black.

And, of course, we've got the visual plot of the equity curve as well.

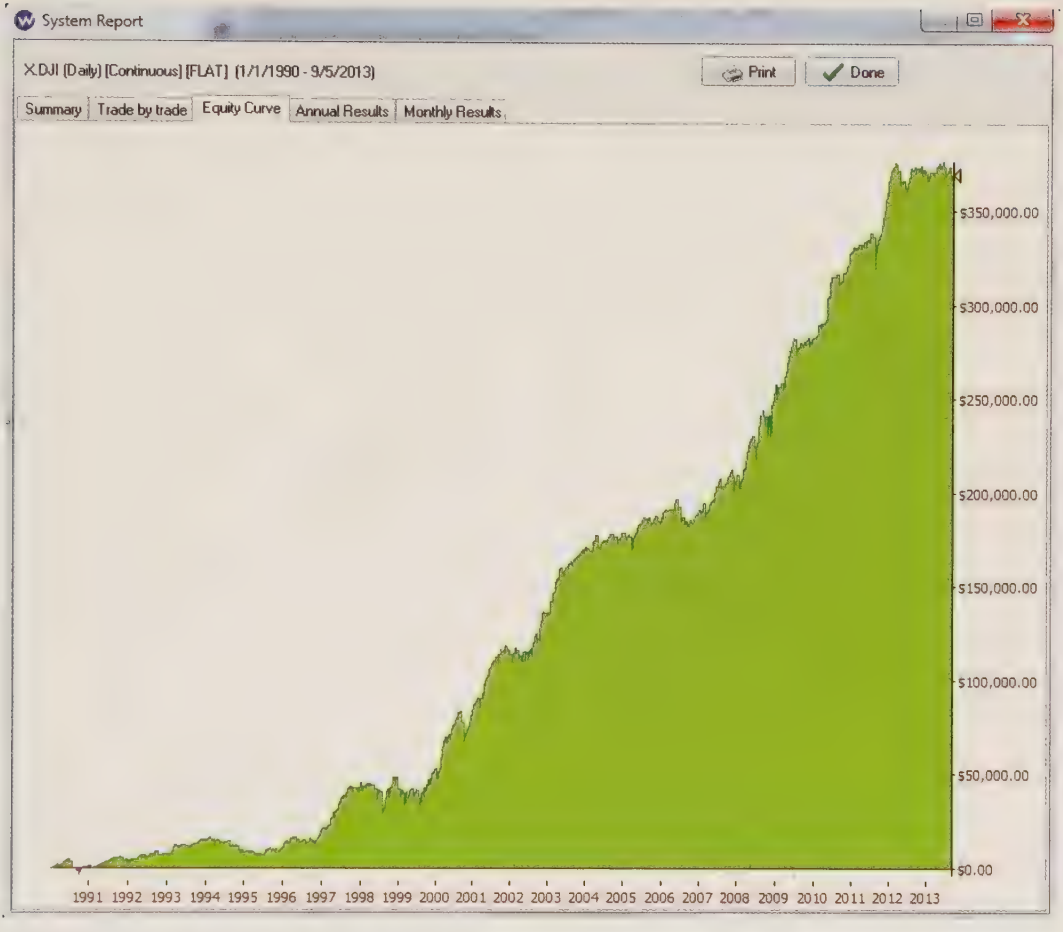


Figure 7.23: System Combined Exit Equity Curve

Although it sputtered a bit in the early part of the 1990s like the others did, it's been great for the last 15 years. If you compare Figure 7.23 to the 2nd profitable close equity curve (Figure 7.20) we just looked at, you can see how the equity line has gotten a little smoother in general. That's the advantage of trading three systems together.

Our total profit was $369,686 and our drawdown was $31,788, so when we take the ratio, we get 11.62, which is better than both the original version, as well as the 2nd profitable close implementation. You get a bigger bang-for-your-buck trading this way, although it does take a larger account to pull it off.

Back in Las Vegas at the PowerUser Conference, I also talked about approaches involving implementing a break-even stop after a certain time, which transformed this into more of a trend-following approach with lower accuracy but higher win/loss. Depending on your preferences, you can take almost any signal and give it a push towards your own trading style simply by using different kinds of exits. And if you're creative about it, sometimes you can merge two different methods together like we did in our last example, like joining an accurate model with another that has a high win/loss ratio, to let strengths and weaknesses balance out. This sort of thing is a great place to spend some time when developing your own systems, especially in light of diversifying across parameters in order to increase system stability.

I'm not going to go through any examples of adding stop-losses, etc, to this approach, and you can do that yourself if you want to see it. I can tell you that the results when doing that sort of thing mirror what we saw in Chapter 6, with the result that this system doesn't trade well with a stop.

Having said that, I should clarify something. Fixed dollar stops tend not to work, as we can prove through testing, but we actually do have a stop built into this system by the nature of how it is set up. The Moon will make one revolution in about 28 days, and we are going to get four signals during that time, two buy signals and two sell signals. Because of that, we have a built-in dynamic stop mechanism in this system, because we can only hold a trade for 7 days before we have to reverse. So although there isn't a hard dollar stop involved, it would be impossible for this system to buy at the top of a bull market and hold it all the way to the bottom of a crash. Time itself will make the system take action, so there's no way an account would get wiped out in one trade, which is the usual rationale behind putting a dollar stop on in the first place. Yes, you can lose a big chunk on one trade, and the reversal method once got caught for a $9k loss as you can see in its system report. But again, if you focus on the account itself, and not the individual trades, you would not have been trading enough size for even that loss to represent a deadly percentage decline when viewed at the portfolio level.

It's not the single trade losses that you need to focus on, which is the short-term way of looking at the system. It's the maximum drawdown value, and that number needs to be taken into the context of account size. Our three-contract combined system had an intraday drawdown of -$31,788 back in 2011, right after the conference attendees

started using this (of course). Is that a lot of money, or not? The answer really depends on your account size.

If you were over-leveraged and under-capitalized, trying to trade three contracts with a $30k account, yes, that drawdown was huge, and would have blown your account right out of the water. But if you had been using the previous worst case drawdown of $20,198 from 2008, and had calculated how large an account you needed in order to absorb that value and still be fine, you'd have weathered the storm and gone on to recoup all your losses and been up almost $40k. Let's do the math again, since the last time we saw this was back in Chapter 2.

Here we are, right after the 2011 PowerUser Conference, and we've just learned a cool system based on Spherical Astrology that we want to trade. Using the 3-contract combined approach, we see that the drawdown in 2008 was -$20,198. Our personal risk tolerance tells us that we don't want to lose more than 25% on our account trading this system. So first we add a pad to our drawdown number, because we know we are probably underestimating it. Adding 50% to our drawdown value gives us a working estimate for max drawdown of $30,297. We then divide the adjusted drawdown number by our desired drawdown percentage to obtain a required account size of $121,188. Let's round that down to an even $120,000.

Great. Let's trade! We transfer $120,000 over to our futures broker, and start paper trading this because we're cautious and want to make sure we're doing it right. It's doing great, and in the six months after the conference makes a bundle. August is time to go live! So then, Mr. Murphy shows up right on time, and the account is down almost $30k in the first part of the month, right after we started trading. Oh no! It's a 26% drawdown, right at our pain threshold, but it's not really worse than expected, and we can keep trading the system, which we do. It took two months to dig back out of the hole, and the account was up from that point on.

That example is how trading goes. You have to be able to weather those drawdowns, and you have to make sure that they don't wipe you out when they happen, because they are guaranteed to happen eventually. Maybe some people are lucky and can start trading right after them, but my experience is that I always start trading immediately before they occur. I suspect I'm not alone either. If $120,000 is too large an account, then two things can be done. The first option is that you can take on more risk, which

will allow you to trade this system with a smaller account. I don't recommend that. Your second option is that you can trade a different system that doesn't require three contracts to implement. That would be a much better idea. Your primary goal is to keep the account alive, and your secondary goal is to grow it. If it gets blown, it doesn't matter how much money the system makes afterwards, because you won't be there to cash in on those signals. So think in terms of survival first. If you can survive long enough, then you'll be able to eventually implement active position sizing in the account, which will exponentially increase the returns. We'll get there soon. You have to walk before you can run, and that is especially true when trading a small account.

8. Machine Learning

So far, all the systems we've looked at have been based on specific timing ideas that we've had to develop. With the William Tell system, we had to figure out exactly what patterns were dominant in Bonds. With the BCI, we had to figure out how to calculate the markets "ideal" price, and with the Lunar system, we had to develop an entirely new branch of Astrology. All of this takes a lot of work and energy to accomplish, as finding one good idea for an entry trigger usually means having to go through a hundred bad ones. Wouldn't it be nice if we could figure out some other way to find these core ideas for system development? That's where Machine Learning (aka Artificial Intelligence) comes in.

Machine Learning is a branch of computer science dedicated to figuring out ways to make computers come up with solutions to problems on their own, without a human having to explicitly set up a rule set. There are a ton of algorithms designed for this purpose, with more being thought up each year. Not surprisingly, the stock market tends to be one of the areas where people attempt to implement a lot of these approaches.

Markets are an ideal application for a machine learning algorithm because there is a massive amount of financial data out there, including individual instruments, broad market data, international markets, fundamental data, sentiment data, and governmental reports, not to mention the thousands of technical indicators available to modern traders. There's so much data that a human can't hope to understand how it all relates, especially if decisions have to be made quickly for intraday trading. However, a computer doesn't have the same limitations, and can look at lots and lots of data all at once. A good artificial trader can synthesize all of this information and see how it all fits together, providing us with insights otherwise unattainable, which we can then use to improve our trading decisions. At least, that's the theory..!

In practice, there have been quite a few approaches used on financial markets, and I've tried most of them out at one time or another. I've looked at neural networks, fuzzy logic, hybrid neuro-fuzzy methods, genetic algorithms, flocking algorithms, and quite a few of my own home-grown applications. I've found that most of them tend to have one thing in common – they are great at producing systems with incredible looking in-

sample equity curves that blow up in your face the minute you attempt to trade real money with them. You haven't experienced a system meltdown until you try trading a neural network with real money – talk about a negative equity curve. Yikes!

For that reason, these approaches seem to show up, gain a ton of short term popularity in the trading community, and then disappear. Most people won't try an approach a second time if they lost money with it on the first attempt. But, despite a lot of the failures, a few of these approaches actually are able to create robust trading methods if applied properly, and I wanted to discuss two of them in this chapter.

Hive Technology

The first algorithm I want to discuss is Hive Technology™, which was developed here at Wave59 over a period of years back in the early 2000's. Hive Technology originally started out as a way to overcome the issue of parameter migration, which we've seen quite a few times so far in this book. One way to solve for parameters moving around is to create a bunch of systems that each use different parameters, and then merge them all together into one master approach. This allows us to leverage the Law of Averages, and attain results in live trading that are much less volatile than what we see trying to guess at various parameter settings. However, the downside to trading a basket of parameters is that the average, by definition, tends to produce results right in the middle of the pack, and never generates close to what we see with the optimal parameters. These types of systems are safe, and make money, but tend not to be spectacular. If they were cars, they'd be the cars that start every morning no matter what the weather, never break down, and always get you where you're going, but don't go very fast, or look very cool, and are boring to drive.

Hive Technology was originally developed as an attempt to improve on this problem. Rather than trading the average of a bunch of systems with different parameters, Hives attempt to discover which particular parameter sets have a better chance of out outperforming the pack in the near term, and use that information to create a different kind of aggregate system. When using a Hive, the results are that the system usually performs as if it had been using parameters in the top 25% or so of all the possible options, which is a significant improvement over the average, which tends to be in the middle.

Hive Technology was good enough at this particular task that it eventually evolved into its own full-blown machine learning algorithm, which takes a bunch of market data inputs, and finds a way to turn those into a mechanical system. The huge feature that sets Hives apart from other machine learning algorithms is that they don't go back and forth over the data set thousands of times to develop curve-fit rules. Since its original goal was as a parameter forecaster, it only looks at the data once, and then makes decisions moving forward. There is no curve-fitting or optimization going on when a Hive is applied to a chart, as everything is set up in a walk-forward method to trade on unseen data. Because of that, Hives tend to be *extremely* robust.

When a Hive is applied to ten years of data, and generates a profit, that profit has been earned on an out-of-sample data set, at least from the Hive's perspective. What you'll find is that when you then take that same Hive and run it forward into more unseen data (from the developer's perspective, as trading begins), the equity curve tends to maintain almost exactly the same characteristics and slope as was seen in testing. It's a huge advantage over other machine learning algorithms, and for me, was a breath of fresh air after having to deal with all those finicky neural networks.

There is a lot of information available at wave59.com regarding Hive Technology, as we're very proud of the algorithm, so I won't repeat a lot of the theory or implementation rules here. You are welcome to check out the various videos on the site if you want to review how to go about building a Hive. What I did want to do is share a Hive that I built a few years ago, so we can take a look at a machine learning algorithm that's been in the wild for a while.

Back in December 2010, I was invited by some Wave59 users to give a talk to their trading group in Chicago. I was visiting for other reasons, and always like to hang out with Wave59 people, so I put together a presentation and went to speak to the group. The second half of my talk was about Hives, and I went through an example and built a Hive and a modified Hive to trade the Dow, which I then gave out to the group afterwards. One of the organizers ended up taping the whole thing and posted it on YouTube, so you can probably find it there if you're interested.

Anyway, the purpose of this particular Hive was to show how useful a well-designed machine learning algorithm can be when used for trading. The only indicator given to the Hive was an adaptive moving average (Wave59's "AMA" function in QScript), and

then it was let loose to trade the Dow. Remember that Hives were originally designed to be parameter surfers, and the best hives like to look at an entire range of indicator parameters, rather than just one. In this case, I let the Hive look at every average between a 2-bar average through a 50-bar average. That's all the information it got, and it had to make all its trading decisions based on just that.

Figure 8.1: Chicago Hive Chart Indicator

When applied to a chart, Hives look like simple momentum indicators. The red line shows how bullish or bearish the Hive is. If the line is above zero (the horizontal blue line), then the Hive wants to go long, and if it's below zero it wants to go short. Like our BCI system, Hives are reversal systems, always in the market, and tend to exhibit all the classic characterizes of the mean reversion approach.

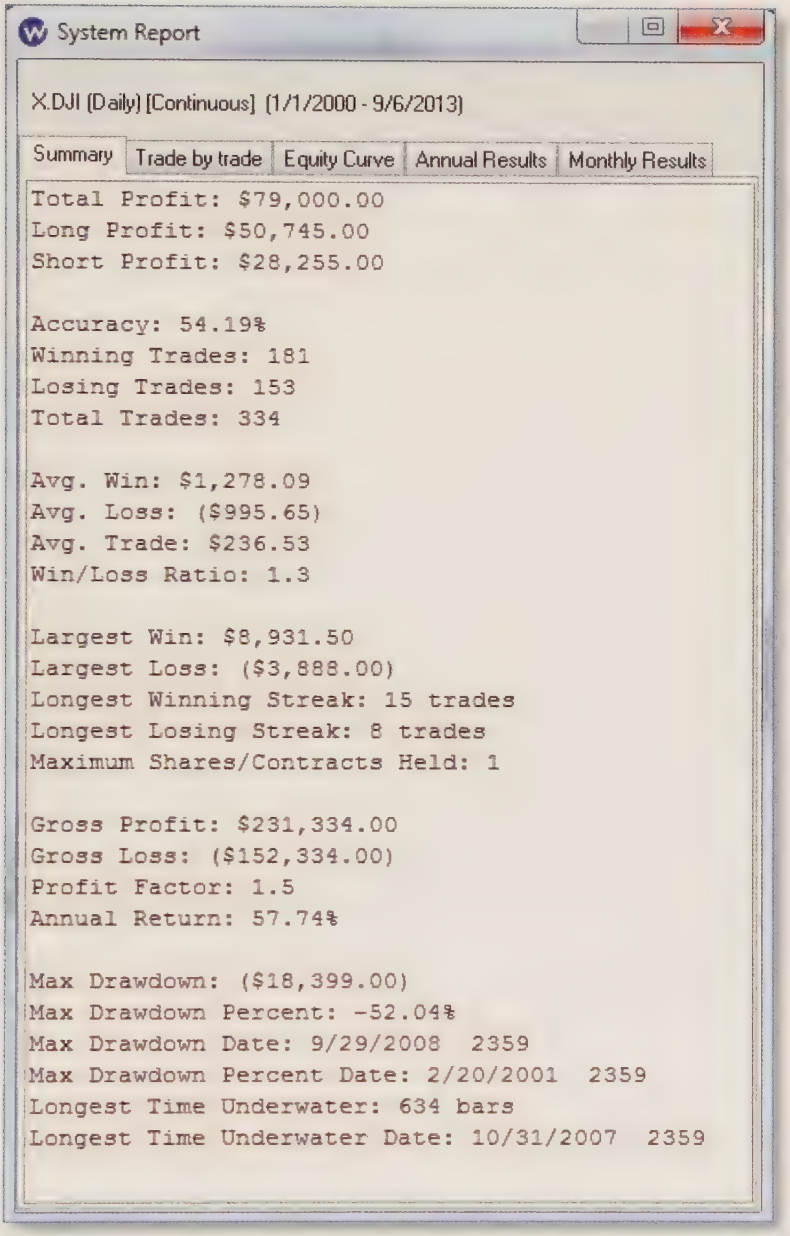


Figure 8.2: Chicago Hive System Report

Since 2000, this Hive has generated almost $80k in profits trading the Dow. It has slightly above average accuracy, and a win/loss ratio above 1. A healthy average trade value, and a slightly large drawdown are also pretty typical of its mean reversion roots.

Figure 8.3: Chicago Hive Equity Curve

Figure 8.3 shows the equity curve. It's a little bumpy, but remember that all the Hive has to work with here are moving averages, which usually result in a different sort of equity curve, especially when considering the direction! You can see how the Hive has performed since the time I first shared it. Also notice Mr. Murphy's appearance in the middle of 2011, right after when someone from that event would have started trading this live.

As I mentioned, I also released a variant of this system, which fits in well with our discussion so far. As in the last chapter, the equity curve above represents a view of our

raw edge. Before, our raw edge was due to a reversal system based on the Moon-Sun ratio. Here, it's based on the output of the Hive. In either case, it gives you a baseline to work with to develop your system further.

In the last chapter, we improved upon our base method by changing the exits around and implementing the Nth-profitable-close exit approach. That would probably work here as well, but another method was used to implement entry filters along the same lines as what we did with the short filter in the Lunar System. Rather than just going long when our signal line crosses above zero and short when it crosses below, we can force the system to try to enter at relatively better prices. When a buy signal is generated, we will only take the signal if the close is less than the close two bars ago. Similarly, when a sell signal is generated, we only take the signal if the close is higher than the close two bars ago. So we sell when the market is moving up, and we buy when it is moving down. Additionally, we take profits earlier as well. So if we're long and the market generates a close above the close from two bars ago, we exit. The same is done on the short side.

This results in a system that uses the Hive's signal as a trend indicator, then looks to trade pullbacks against that trend, with quick profit takes. Everything is based on the relationship between the close of the current bar and the close two bars ago, so these trades will be significantly shorter in duration than what we saw earlier.

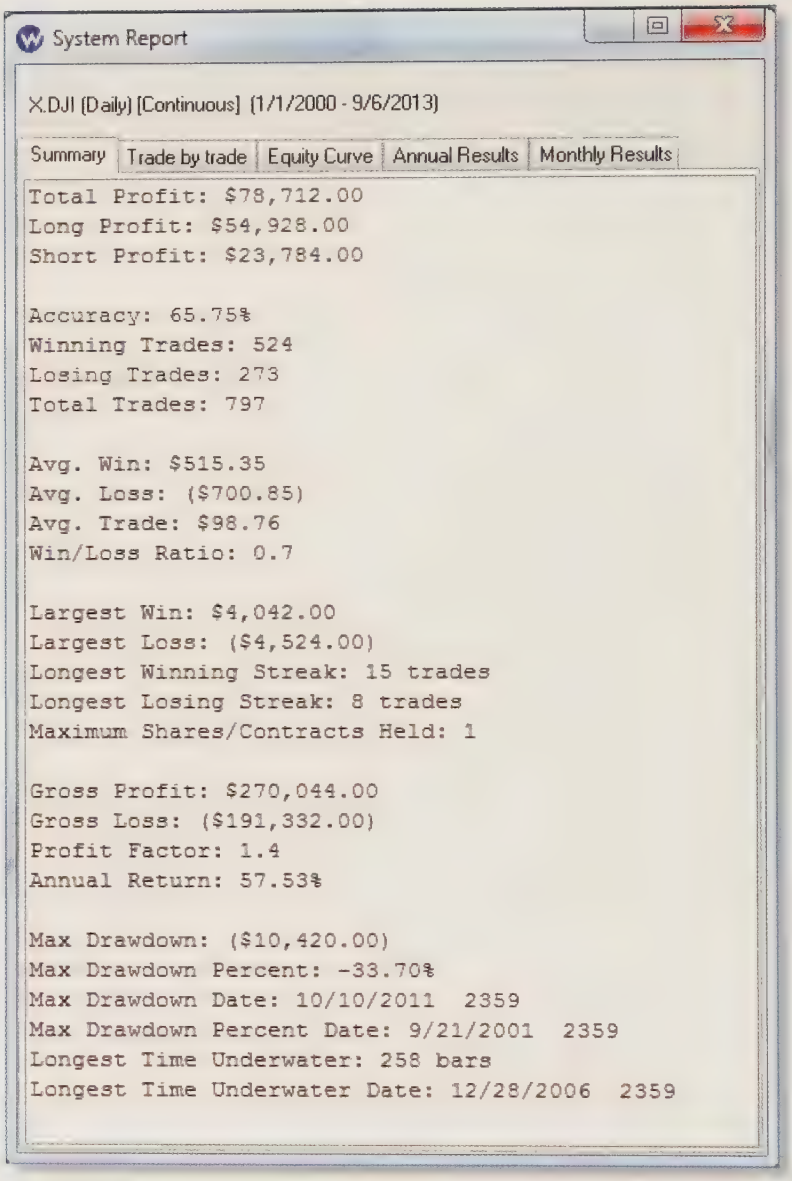


Figure 8.4: Chicago Hive Modified System Report

You can see that the system makes almost the exact same amount of profit as before, but it does so in a completely different way. Rather than being a typical mean-reversion system, it is now taking on the characteristics of an accuracy-based method. Accuracy has increased from 55% to 65% while the number of trades doubled. As expected,

average trade and win/loss fall. Most importantly, notice the drawdown number. It is down to -$10,420, a 43% reduction! What that means is that we earn the same profit, for almost half the risk. Said another way, we could earn almost *twice* the profit for the same amount of risk by trading the modified method instead of the original.

Figure 8.5: Chicago Hive Modified Equity Curve

The equity curve for the modified approach is shown in Figure 8.5. It looks a lot "fuzzier" than the last equity curve. That is because there are twice as many trades going on, which adds a lot of detail to the plot. You can see how this method has performed during the 3 years it's been in the wild.

Hives are definitely a Wave59-centric approach, as you need special internal functions in order to run them which aren't available in the public domain. The scripts for both of these implementations are found in the appendix, and can also be downloaded from the

Wave59 script library at www.wave59.com/library if you don't want to have to type it all out. Just search for "Chicago Hive" and you'll find them.

I've always been surprised that more people haven't taken advantage of Hive Technology. The systems produced tend to be quite robust, as well as profitable. The private fund I'm involved with that I mentioned in the introduction uses Hive Technology extensively, and that system has been cranking away like nobody's business. If you're going to explore the machine learning realm, you could do a lot worse than spending your time working with Hives. They won't generate nearly as pretty equity curves as you'll find with neural networks or other massive optimizers, but they actually work when used in live trading, which is not a feature those other techniques can really lay claim to.

Genetic Algorithms

Genetic algorithms have been around for decades now, and have proven their worth in countless applications in many diverse disciplines, including trading. A **genetic algorithm** is a method based on ideas in evolution and natural selection, that allows computer scientists to quickly find solutions to problems that would otherwise be unsolvable due to the nature of the problem involving so many possible combinations of various parameters that testing each one would result in consuming hundreds of thousands of years of computation time, even with the fastest computers in the world.

There are a couple of really cool things about genetic algorithms. The first thing is they are based on models that you would normally find when studying biology, which isn't something you'd expect to see in a computer algorithm. Second, they can solve problems without really even knowing what they are trying to solve. You just need to give them a way to tell how "good" a proposed solution is, and they don't need to know anything else about what you are trying to do. Because of that, you can use them to solve for almost anything, from simple math problems, all the way to insane weather modeling problems.

I'm going to walk through the basics of how a genetic algorithm works. It's not difficult to understand, and if you've never seen this before you'll be impressed. You really only need to know the technical details behind them if you plan to program one, but if you

are going to use the systems presented shortly, I think it's a good idea to have a general understanding about how they came to be.

Being based on ideas of evolution and natural selection, a genetic algorithm creates a population of many individuals, each with slightly different genetic information. A solution to a given problem is encoded within each individual's genetic makeup, and using a fitness function, the genetic algorithm is able to evaluate how fit each member of the population is which allows it to rank each individual within the population. More fit members of the population are allowed to mate with each other, and their children are new solutions that are made up of a combination of both parent's DNA, just as in the real world. Less fit members of the population have difficulty mating, and are often killed off without being able to pass their genetic information along. As such, there's a survival-of-the-fittest race going on inside the algorithm, where individual members of the population strive to survive and pass along their genetic material in what can be a very hostile environment.

Let's run through a simple example, and see how all of this works in practice. Our first step is to find a problem that we want our genetic algorithm to solve. I want to keep this simple, so I'm going to use the following problem: I'm going to let the genetic algorithms control all the light switches in the room I am now in, and I want it to figure out how to turn all of the lights on.

The next thing we need to do is then figure out a way to encode the various variables to this problem into a chromosome. From where I'm sitting, I can see 9 light switches in this room. Therefore, our chromosome will include 9 numbers (or genes), like this:

110000001

The chromosome mentioned above is a 9-digit number made up of 1's and 0's. Each position in that number corresponds to one of my light switches. If the position in the number has a "1", it means that the switch should be up. If the position has a "0", it means the switch should be down. In the case of the number shown above, we see that this represents a solution where the first two switches and the last switch are all up, and the rest are down.

So you can see how one chromosome represents a complete solution to a particular problem, encoded in numbers. The genetic algorithm itself doesn't need to know anything about the nature of the problem, it just needs to know what the chromosome looks like. Now let's start the process of evolution.

What happens first is that an initial population is generated, with many different members. This is done randomly. In real problems, populations can be quite large, but in our case, let's assume a very small population of only 4 members. Specifically, our population looks like this:

Population Member	Chromosome for that Individual
1	110000001
2	100000010
3	000111100
4	101000101

Table 8.1: Genetic Algorithm Example Generation 1

We now have our initial population. The next step is to evaluate the fitness of each of our individuals. This is done with a fitness function, and is the way that the genetic algorithm ranks the members of the population. In our case, we are trying to get the lights turned on, so we can simply add up all the numbers in our chromosome and use that as our fitness function. The more lights get turned on, the brighter it will be in the room, which is a good way for the algorithm to measure how well it is doing.

Let's now figure out the fitness for each of our population members:

Population Member	Chromosome for that Individual	Fitness
1	110000001	3
2	100000010	2
3	101000101	4
4	000111100	4

Table 8.2: Genetic Algorithm Example Generation 1 Fitness

We can see that the best solutions are given by population members three and four, both of which have got four light switches flipped. Individual number two is the least fit member of the bunch, with a score of only 2.

If this experiment were happing in nature, the result would likely be that the first two population members, being the least fit, would end up dying or getting eaten by predators. Conversely, the most fit members would survive, and be able to pass along their offspring to a future generation. Our algorithm models this when it generates the next generation of the population. As the first generation dies off, four new slots are opened up in the population, and these get filled with the children of the most fit members.

In our case, members three and four represent the top of the pecking order, so those two chromosomes become the seeds to create the next generation. This is done by merging the genetic structure from the two parents in a way that creates a child. The easiest way to do this is to randomly select a point to split the chromosome, and then to take everything before that point from one parent, and everything after it from another. Assuming that we selected a split point at the 5th gene in our sequence, we would create a child chromosome as follows:

Gene Number	Parent 1	Parent 2	Child gets Gene from Which parent?	Child Chromosome
1	1	0	Parent 1	1
2	0	0	Parent 1	0
3	1	0	Parent 1	1
4	0	1	Parent 1	0
5	0	1	Parent 2	1
6	0	1	Parent 2	1
7	1	1	Parent 2	1
8	0	0	Parent 2	0
9	1	0	Parent 2	0

Table 8.3: Breakdown of Crossover at Gene 5

I've arranged everything in the table above. Our chromosomes are 9 numbers (genes) long, and we have two parents. These come from the two fittest members in the previous table. We randomly selected a split point at the 5th gene, so the first four genes get contributed from our first solution, and the rest get contributed from the second solution. This creates a brand new solution, which is 101011100.

We then repeat this process for each spot in our population of four, randomly selecting different ways for the most fit members of society to swap their genetic information. We only have two members to work with in our tiny example population, but in larger solutions there would be lots of fit members all swapping genetic material with each other to create the next generation.

We have already created our first child, after a swap at place 5. We'd generate three more children, swapping (randomly) at places 2, 7, and 9. That would give us the following result:

Generation 2		
Population Member	**Chromosome for that Individual**	**Fitness**
1	101011100	5
2	100111101	6
3	101000100	3
4	101000100	3

Table 8.4: Genetic Algorithm Example Generation 2

Usually during the process of genetic propagation, mutations in the child chromosome also happen. This gives the genetic algorithm the ability to search around in the solution space to find combinations that might not be initially present in the original chromosomal data. In our example, mutation has occurred to the final gene in population member 2's chromosome, randomly flipping it from a 0 to a 1. This replicates the same way that nature experiments, where a black moth might suddenly find its way into the population when all the other moths are white. Some mutations don't help an individual's survival while other times mutations allow whole populations to make huge leaps forward. If the mutation is not helpful, it will tend to be dropped from the population through evolution, and if it doesn't hurt, or is helpful, it has a chance to stick around for the benefit of future generations.

Now that our second generation has been set up, and our algorithm knows which members are the most and least fit of the bunch, the process repeats. Members one and two are the most fit, with fitness values of 5 and 6. Therefore, they survive, while the poorly functioning members of society drop off. Running through the exact same procedure, and crossing the first two chromosomes at randomly selected gene locations of 2, 4, 7, and 8, we come to our third generation.

Generation 3		
Population Member	**Chromosome for that Individual**	**Fitness**
1	100111101	6
2	101111101	7
3	101011101	6
4	101011101	6

Table 8.5: Genetic Algorithm Example Generation 3

You can see how the entire population is getting better and better at turning on the lights in this room. Individual two already has seven switches turned on. If we kept going, eventually we would come to a place where all of the lights were on, which is the problem we set out to solve. Notice how all of this was done without the genetic algorithm really having any idea about what the specifics were behind the problem itself. It only has its chromosomes and feedback on its performance through the fitness function. As long as we can encode our problem into a similar chromosomal structure, and provide a fitness function, we can apply genetic algorithms to whatever problem we want to solve.

In the realm of trading systems, you should already be seeing applications for this technology. The most common implementation for genetic algorithms is in optimizers. So far in this book, we've been making use of a brute-force optimization module in Wave59 which tests every single possible combination of a set of parameters, and displays the results. This works well for systems with a small number of parameters, but not so well otherwise.

Imagine a system based on the crossovers of two moving averages. To look at the results of this system through our optimizer, we would let it run through a set of parameters for the shorter average, as well as a set of parameters for the longer average. If we test 20 combinations of the short length, and 20 combinations of the longer length, that gives us 400 different solutions that need to be solved. If this is a massive data set, and our system is really complicated, it might take 5 seconds to run one system, which means that this test would take us around half an hour. But what happens if we decide to add two more moving averages, which we would use to trigger a secondary exit? Now we have to test 20x20x20x20 = 160,000 combinations, which would take over 9 days! That's a long time to leave the computer running, and you're

definitely going to have a problem if you have some other parameter that you are optimizing at the same time.

As more and more parameters need to be optimized, a brute-force method quickly becomes impractical. There just isn't enough time to check all of the possible combinations. In that case, a genetic algorithm can be used instead. If each gene represents one particular parameter solution, then it becomes very simple to generate a population of solutions and let the genetic algorithm figure out the best parameters. The fitness function would simply be how much profit the system generated when run using the solutions encoded in a particular chromosome. The genetic algorithm doesn't know anything about trading specifically, but the fitness function tells it how well it's doing, and it goes about finding better and better solutions in the same way it figured out how to turn on all the lights in my room.

In the past, I have used genetic algorithms for all sorts of problem solving, all the way from finding parameters to various systems up through solving for weights in neural networks. In the past few years, I've applied them in even more interesting ways, which I'd like to share here.

Since a genetic algorithm doesn't need to know anything about what it's really solving, we really aren't limited by what we can throw into it except by how difficult it is to encode our problem into a chromosome. But as system developers, what is it that we would really like for our algorithms to do? Are we content to use this technology just to solve our moving average lengths? What about widening our view, and giving the genetic algorithm full reign over the entire development process? What about letting the genetic algorithm actually write our source code?

Implementing this was one of the more interesting projects I've been involved with. QScript, Wave59's purpose-built scripting language, is uniquely designed to leverage some very powerful trading functions when building mechanical systems. So rather than a chromosome containing various parameter sets, I built a genetic algorithm where the genes in a chromosome actually contained lines of QScript source code. Each chromosome itself then represents a complete computer program, which runs within Wave59.

Remember that a genetic algorithm doesn't need to know anything about the actual problem itself, aside from the need to access a fitness function to tell how it has been doing. It also doesn't need to know how the chromosome itself gets decoded into a solution. It only needs the chromosomes and the fitness function, and it will then go and figure out good solutions to whatever that particular problem is. The fitness function I used was a profitability measurement when running the script on a daily ES chart.

As I watched the process, what happened was that the genetic algorithm initially tried a bunch of really stupid things, but as time went on, the solutions got better and better, until all of sudden the numbers were above what I was able to generate myself using the same tools. I remember the point when I realized this was going to work, and it's a funny story. For years, I had used momentum indicators to find trading opportunities, and I've gotten good enough at spotting momentum divergence that I can do it in my sleep. Heck, I've written books and articles about doing just that thing. So, imagine my chagrin when my genetic algorithm handed me a system that said it was better to trade against momentum rather than with it. In other words, momentum works better when you flip it upside down! All the years that I had been working with that indicator, and it was like the computer said "You've got the page upside-down, dummy!" Whoops.

This was the first system that the genetic algorithm generated, so let's look at it. This has been sitting in front of my face for over a decade now, but I just never saw it because I had the indicator line upside down on my charts. The rules are extremely simple:

> *Buy when UltraSmooth Momentum crosses under zero*
> *Sell when UltraSmooth Momentum crosses over zero*

That's it! Use UltraSmooth Momentum (USM), but do the opposite when it crosses over zero. You'll note that this is exactly the reverse of what every indicator trader has been taught.

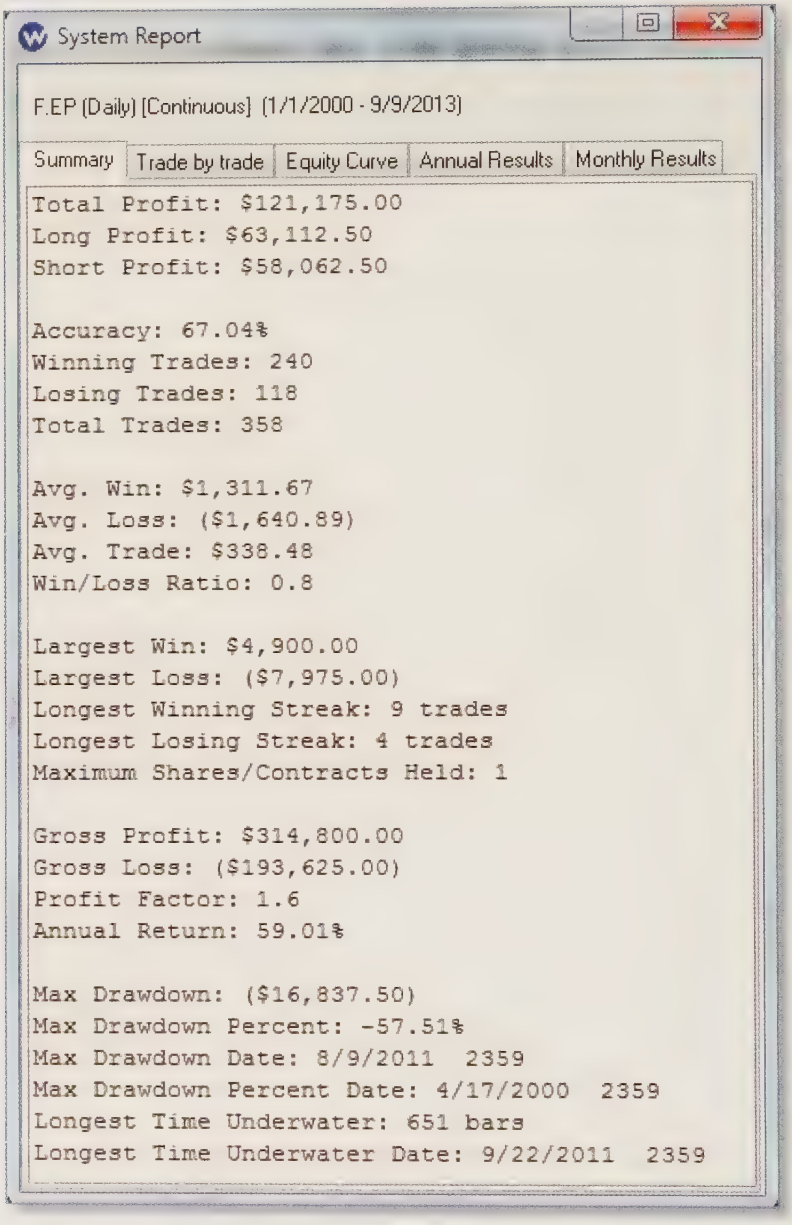


Figure 8.6: Inverse Momentum System Report

Figure 8.6 shows the report for the Inverse Momentum system running on the Emini since 2000. It's almost 70% accurate!

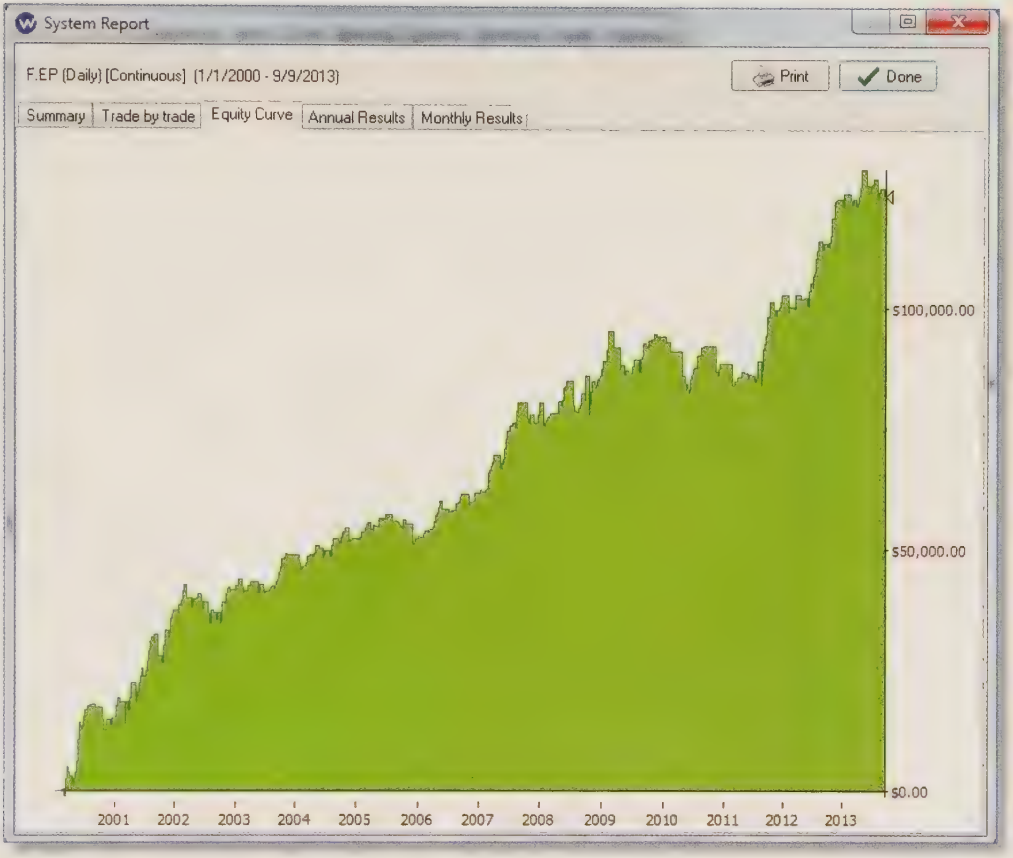


Figure 8.7: Inverse Momentum Equity Curve

The equity curve is pretty good too. It had a little trouble around 2009-2011, but for the most part we see a good upward slope. I always knew that UltraSmooth Momentum was a step above the typical indicator in technical analysis, and the genetic algorithm proved it to me. I just had been misinterpreting how to use this tool. You don't have to sit there and watch for divergences and look for correlations with other indicators to find good trades – you can just blindly trade the zero line crossovers and not even look at the chart if you don't want to, and you'll make money with this one indicator.

What about the parameter issue? The system shown uses a momentum length of eight, meaning that the momentum line is calculated by looking at price action over the last eight bars. We need to make sure that our genetic algorithm isn't using a silver bullet

parameter here. Let's run this system on the optimizer, and check a range of length parameters.

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=2	$151,375.00	65.54%
2	#1#=3	$114,225.00	65.99%
3	#1#=4	$71,800.00	63.49%
4	#1#=5	$62,150.00	62.65%
5	#1#=6	$70,100.00	63.39%
6	#1#=7	$114,725.00	65.05%
7	#1#=8	$123,150.00	67.32%
8	#1#=9	$98,175.00	66.67%
9	#1#=10	$90,450.00	70.53%
10	#1#=11	$79,050.00	70.00%
11	#1#=12	$26,425.00	65.77%
12	#1#=13	($675.00)	64.29%
13	#1#=14	$3,275.00	64.22%
14	#1#=15	($20,225.00)	61.93%
15	#1#=16	($2,675.00)	57.84%
16	#1#=17	$12,200.00	60.12%
17	#1#=18	($2,500.00)	60.61%
18	#1#=19	($1,662.50)	62.26%
19	#1#=20	$6,800.00	62.42%

Figure 8.8: Inverse Momentum Look Back Length Optimization

Figure 8.8 shows the system running using momentum length parameters from two through twenty, which is about the usable range for this indicator. 75% of all parameter choices made money, and the losers were all longer term lengths. This seems to be a robust system that should be traded with short momentum lengths to try and catch fast moves. The value of eight we were using does seem to be better than the surrounding parameters, but check out the results using a super-fast two-period momentum. Shorter is definitely better here.

At this point, after a day or two of running my genetic algorithm system designer, it was able to see a very simple relationship that I had missed despite looking at this indicator for a really long period of time. Maybe the genetic algorithms were better than I was at

finding these relationships..? So for the next six months or so, I let the genetic algorithms loose, and observed them as they built system after system. Some of the systems were clearly over optimized, and way too brittle, but others, like the Inverse Momentum approach, were great.

I set up a bank of computers, each evolving a population of trading systems, and would let the computers crank overnight. I'd come in the next morning and spend most of the day looking through what the computers had found for me, and ended up building quite a library of impressive systems. I'd like to share some of these systems with you in the next few pages.

Here's an interesting one:

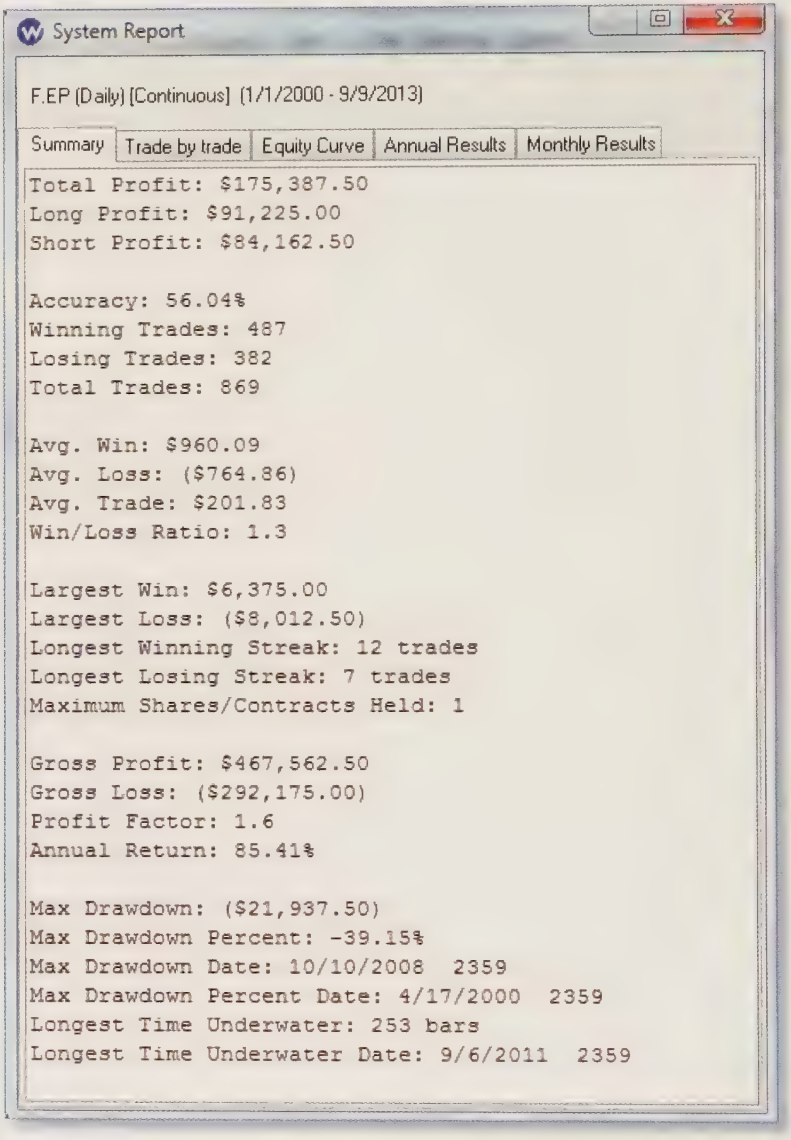


Figure 8.9: Genetic Algorithm Built System #1 System Report

You can tell that the genetic algorithm likes to build reversal systems. We've got almost 56% accuracy, and a win/loss ratio of 1.3 trading the Emini since 2000. Total profit is $175k during this time, which is a HUGE number. You won't find many developers that can build a system that profitable by hand.

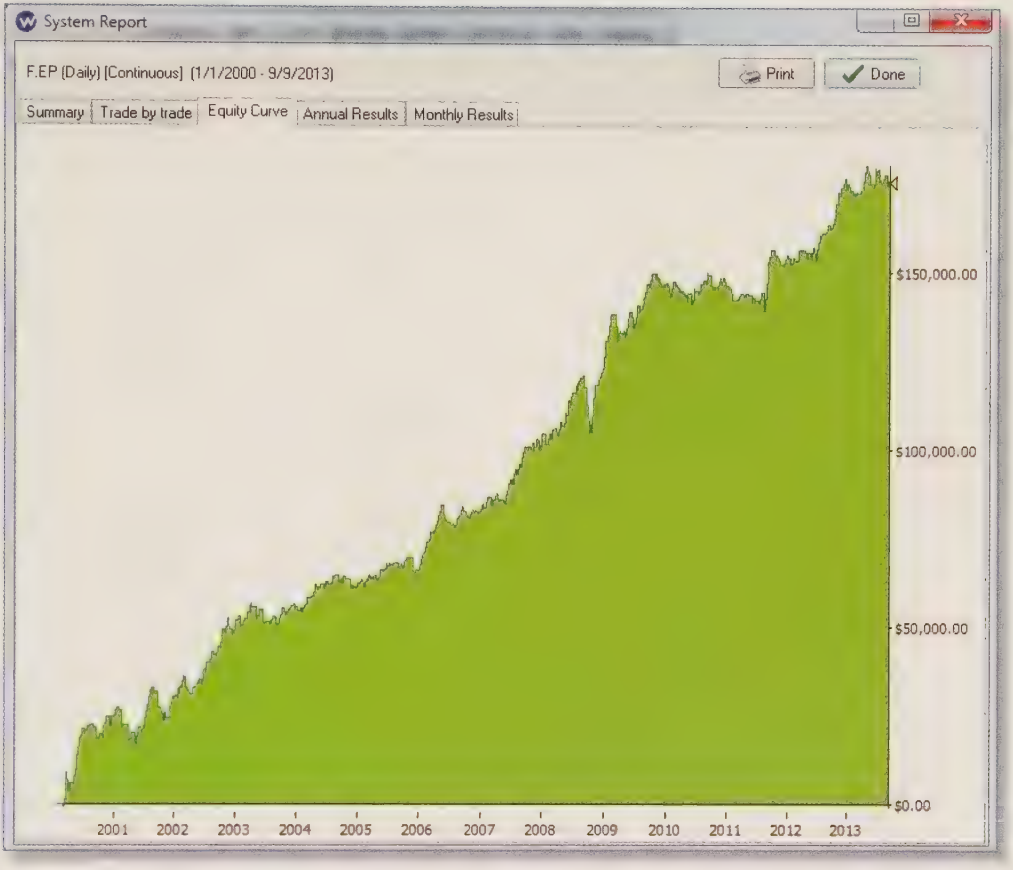


Figure 8.10: Genetic Algorithm Built System #1 Equity Curve

This is a great equity curve too. The losing periods are very short in duration, which results in very stable account growth. This particular system popped out of the genetic algorithm in the first few months of 2012, so it's got about a year and a half of live results under its belt at this point.

So how does it work? I don't know..! I can tell you what goes into it, and I can see how the genetic algorithm merges it all together, but I really don't understand what it's doing. Computers don't view the world as we do – they think purely in numbers and math, and it can be extremely difficult to work out what they are doing.

All I can say is that this is a refinement of the simple Inverse Momentum system we just looked at. The genetic algorithm uses six bar-to-bar relationships, such as the close

minus yesterday's high, and has basically created a new signal line by merging our familiar momentum tool with its own calculations, with a little trigonometry thrown on top. The result is that it outperforms every one of the parameter combinations we optimized for in the Inverse Momentum system, and has produced a much better system.

Here's another variation on the Inverse Momentum theme:

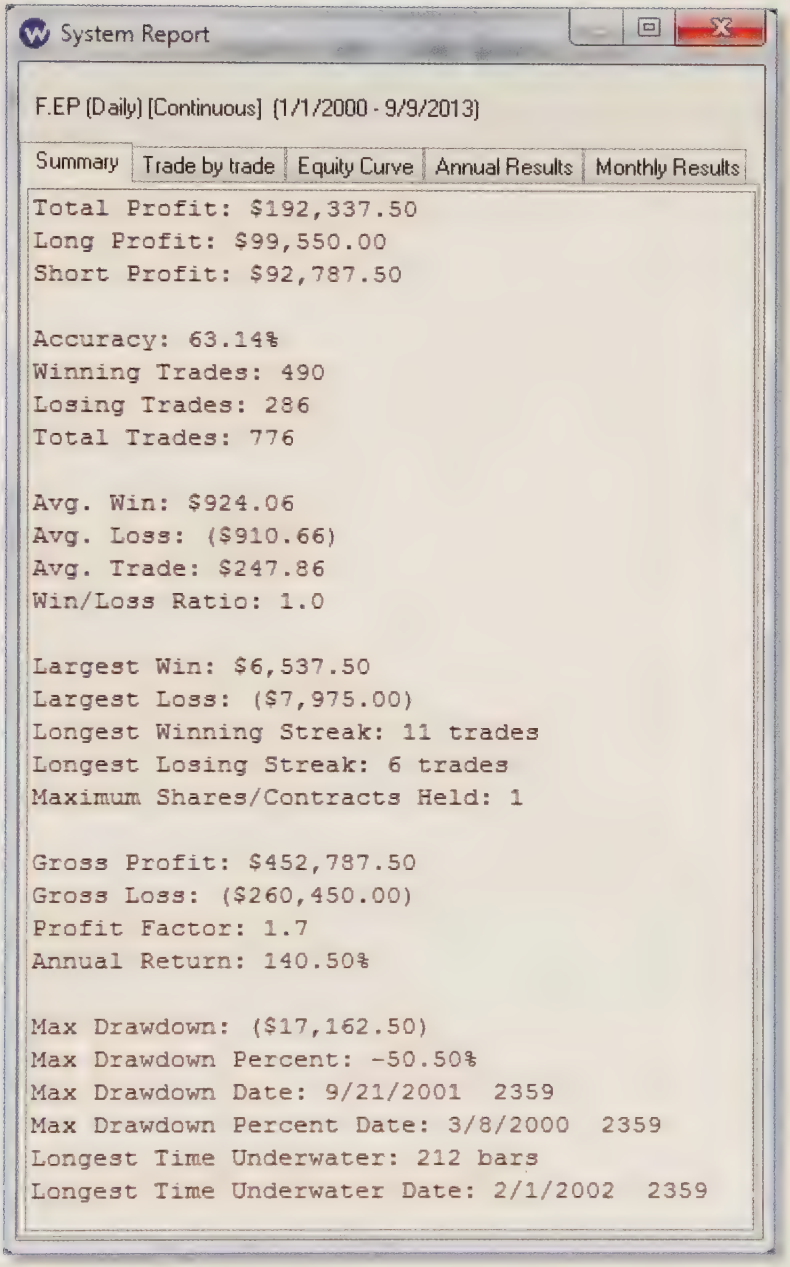


Figure 8.11: Genetic Algorithm Built System #2 System Report

You can see that the genetic algorithm has done an even better job at making the base Inverse Momentum approach into something more. We've gained $17k in profits,

reduced drawdown by $5k, and improved accuracy to 63% when compared to the previous system. The changes might seem small at first, but these sorts of improvements will have a dramatic impact on a system's performance in real life, especially when we take position sizing into account.

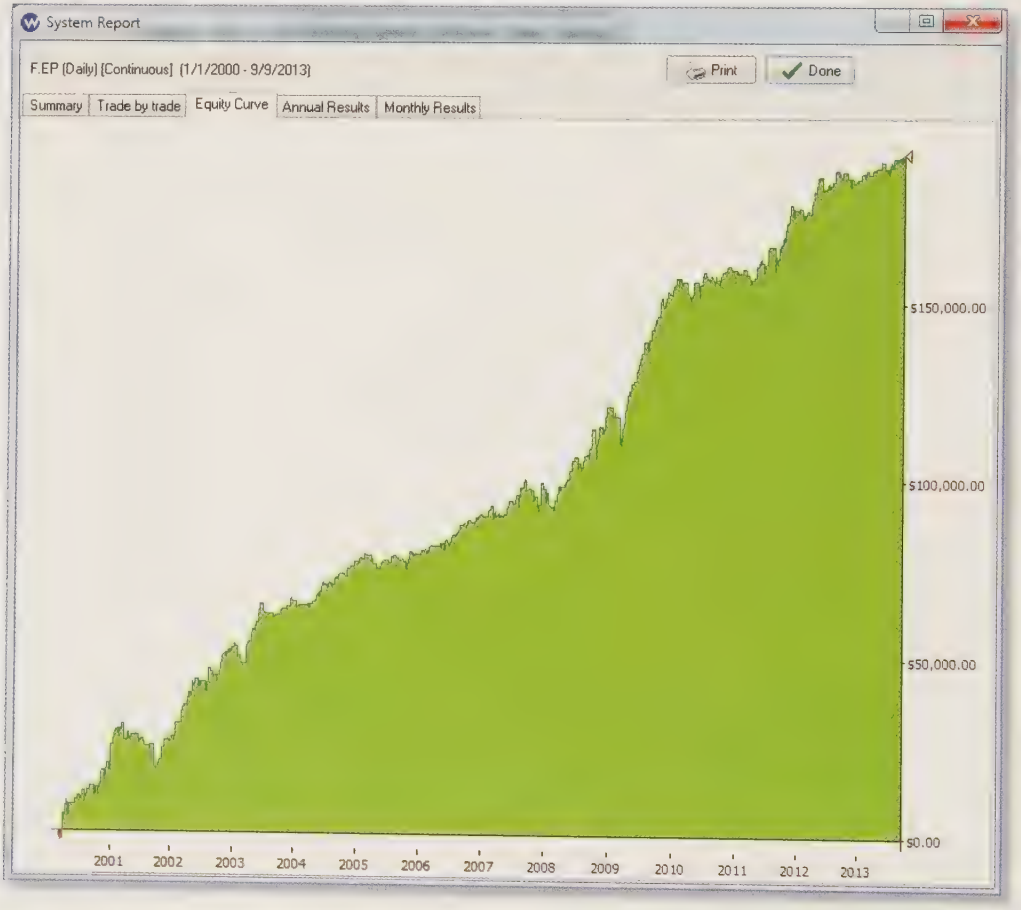


Figure 8.12: Genetic Algorithm Built System #2 Equity Curve

Notice how much better this system handled the bumpy 2010-2011 period, especially when compared to the original Inverse Momentum system we looked at earlier. Again, this was evolved in the first couple months of 2012, and has been on a tear through the real time trading period since then.

Since this particular approach has a relatively small instruction set, I wanted to go through it just to give you a flavor of how computers "think." What we're going to do is take three different UltraSmooth momentum curves, and combine them together. We're not going to be doing averaging or anything like that though. That's too simple, and simple math is for humans. ☺

To keep this clear, I'm going to use the following notation for our three momentum indicators:

USM2 = 2-period UltraSmooth Momentum
USM10 = 10-period UltraSmooth Momentum
USM18 = 18-period UltraSmooth Momentum

The formula that we are using is:

$$(\frac{1}{USM18})^2 - \left(\frac{1}{USM2}\right) > \sin(USM10)$$

If that expression is true, we're buying. If it's false, we're selling. You can see what I mean when I say that I don't really know what the genetic algorithm is doing. Obviously it's making a comparison between USM10 and the other momentum indicators, implementing a sort of cross-over system, but why does USM18 get squared, and what's with the sine function? Machine learning can show you the solution to a problem, but it can't explain the solution to you. You have to be able to work it out on your own, or just accept the answer and use it.

Back when I used to offer William Tell to the general trading community as a subscription service, I used to get a kick out of some of the people who insisted that they needed to know exactly how the system worked before they'd be willing to put any money behind it. The fact is, even if you show the script that implements a system to the average person, that person is still not going to have any idea how the system really works. Out of all the traders using Stochastics in their daily trading activities, what percentage have actually looked at the formula? And out of that percentage, how many people actually knew what that formula was really doing? The fact of the matter is that most people are trading blindly. They are using tools derived from calculations they really don't understand.

So in order to have faith in a tool, the calculation itself isn't always the main thing. What is more important is to go back over previous history, learn how the tool works, and become intimately familiar with its behavior. Then you will gain confidence and will "know" the tool, even if you can't compute it yourself. This is a good thing, because when it comes to machine learning, that's about as close as we humans are really going to be able to get to understanding what is going on, even if the formulas are printed out and shown to us.

The main thing that we have to decide is whether or not the formula seems overly complicated to the point where it might be too curve-fit to past data and likely to blow up in the future. Does the formula above fall into that category? That's a judgment call. Obviously, I didn't think so, as I built it up into a system that I'm running (to be more precise – an ingredient into another system, but we'll get there soon).

Genetic algorithms do have a downside, and are a double-edged sword. They were originally built as massive optimization engines. You can put a thousand variables in there, and the genetic algorithm will go through and find a solution that optimizes all of those variables all at once. But even the best tools can still be misused. If you want to force the genetic algorithm to create a perfect trading system that crushed the market and hardly ever had a drawdown, you can do that, but the result will likely blow up in real time. Ferraris can go fast, but if you're not used to driving in high performance cars, it can be really easy to crash them.

There are a few critical things that separate a good system from a bad one when looking at genetic algorithms. They are:

1) Good inputs
2) Good fitness function
3) Reasonable expectations

Good Inputs

If you allow the genetic algorithm to pick and choose from any old variables, they will oftentimes use inputs in their models that you really don't want them to use. It's as the old saying goes, "garbage in, garbage out." So it's important to give the genetic algorithms (or any machine learning method) inputs to use that really do provide useful information about the problem being solved. UltraSmooth Momentum is the perfect

example. It can be used all on its own to create profitable trading systems, so when we then pass it into a genetic algorithm, we are giving the computer valuable information. This would not be the case with many other indicators available in the public domain, most of which can produce squiggly lines on a chart, but not much else.

Good Fitness Function

As mentioned earlier, genetic algorithms don't actually know anything about markets. They just know how well they're doing by using the fitness function to evaluate the relative performance of various individuals in the population. A good fitness function can steer the evolution towards solutions that result in robust, tradable systems. Conversely, a bad one will steer evolution towards curve fit solutions that blow up immediately when applied to live data. The genetic algorithm will optimize according to whatever parameters the operator specifies, so you have to be careful and build your genetic algorithm correctly.

One of the worst fitness functions to use is pure profitability. We don't really want the most profitable solution, because that is likely the solution that will also be the most curve fit. What we need to do is teach the genetic algorithm to think the same way that we would when evaluating a system report. In the last chapter, we found a Venus aspect that had great profitability, awesome accuracy, and a perfect equity curve. We also threw it away because it traded too infrequently, since we basically didn't trust the numbers. Our genetic algorithm needs to be smart enough to throw that pattern out too. I suspect this is one of the reasons that most commercially available machine learning platforms tend to fail miserably when it comes to producing tradable systems. They are set up to produce curve-fit systems, and so they do.

Reasonable Expectations

Finally, we need to have reasonable expectations about what we are trying to accomplish when using genetic algorithms. If we think that we can push a button and receive the holy grail of trading systems out the other end, we're going to be very disappointed. However, if we set our sights on incremental improvements using inputs that are already proven in other approaches, we can generate all sorts of interesting systems to help in our trading. In other words, we need to use genetic algorithms more as research partners, and allow them to help us sort through the mountain of data that the markets provide.

In this chapter, we've looked at two machine learning applications that I've used profitably in my own trading. If you want to develop your own methods, but have trouble coming up with good ideas to base a system around, this is a great area to explore to provide you with your edge. You're still going to need to go through the process of evaluating the system and slowly building it out as we've been doing in the other chapters, but you don't necessarily have to be able to make market discoveries in order to construct very tradable systems that you can use in your own trading operations. In case you're curious, the genetic algorithm implementation discussed in this chapter will show up in one of the early Wave59 3.0-series builds, the first of which is scheduled for release shortly after this book goes to press.

9. Random Numbers

If you've ever flipped through the pages of any trading magazine, you'll be very aware of how much emphasis is put on entry signals in this industry. Nine out of ten articles focus on some special way to find a trading signal, and authors scramble to outdo each other using various methods that are supposed to increase profitability and get the reader that much closer to being able to buy the exact low and sell the exact high of tomorrow's price range. You can study entry techniques based on classical chart reading, candlesticks, oscillators, fundamentals, inter-market analysis, seasonals, sentiment, market internals, artificial intelligence, cycles, astrology, and numerology, among other things. The list is endless, and each branch of analysis subdivides into all sorts of other branches, each leading to more and more tools that can be used to find those all-important entry points.

But how important are entry signals, in the whole scheme of trading? In order to place a trade, you must know four things:

1) What to trade (your market)
2) How many shares/contracts to trade (your size)
3) When to get in (your entry)
4) When to get out (your exit)

Without the first three items, you can't even place an order, and without all four of them, you don't have a complete trade. Given the way information seems to flow in the trading industry, the fact that all the focus tends to get put on the third item in the list is a good indication that that particular item is actually the least important of the bunch.

We've already seen in previous chapters that the exits we adopt have a huge impact on how the system performs. The exits impact not only the accuracy and win/loss ratios, but the entire shape of the equity curve in general. Exits done improperly can turn a highly profitable system into a severely diminished one, or even an unprofitable one as we saw in Chapter 6. Conversely, exits done properly can turn even a marginal system into something that makes money.

As this is a systems book, and not a theory book, we need to prove all the statements made. Therefore, I'm going to build a system that trades using *random numbers* as

entries. That will completely obliterate the concern over entry techniques from the landscape, and will squarely place the success or failure of the system entirely on the shoulders of our exit technique.

If you've never done this sort of experiment before, it's extremely enlightening. Take random entries on your market of choice and then apply different exit techniques. It will become clear very quickly which exit techniques are helpful, and which aren't. Not surprisingly, the most common exit approaches, namely fixed-dollar stops and fixed-dollar targets, will turn random systems into consistent losers. But there are exits that will do the opposite, and turn those random systems into profitable ones. One such exit is based on the Nth-profitable close technique we've seen a few times already in this book.

Specifically, we will take profits when a certain number of profitable closes takes place. In addition, we will exit the trade at a loss when a certain number of unprofitable closes takes place. The entries themselves will be based on a random number generator, which will tell us whether we are buying or selling. Every 20 bars, we'll place a trade on the daily ES market, and we'll do this experiment 10,000 times, as each run will produce a different set of signals for us to test since they are all completely random.

Running this test through our optimizer produces some very interesting results. For example, taking profits at the 2nd profitable close, and taking losses at the 6th unprofitable close results in 73% of the runs being profitable. The optimization results are way too long to show, but if we average out all the runs, we find the following aggregate results:

System Metric	Average Result (after 10,000 runs)
Profitability	$11,535
Accuracy	68.7%
Average Trade	$56.68

Table 9.1: Trade Statistics for Random Entry Test

As these results are based entirely on random numbers, if I ran this again, I'd get a slightly different result. However, the main point is that, on average, this exit strategy usually makes money. In fact, it makes money in almost three out of every four systems, and that's way better than the odds you'll find in the classified section of the trading magazines!

One of the other things you'll notice is that these systems tend to be quite accurate, and we had an average accuracy result of almost 70%. If you think about exactly what our exits are doing, the reason for this will become clear.

A system like this basically assumes that markets are just noise, especially since we are fine using random numbers as our entry trigger. As such, the only thing we have control over is the exit. If we make it easy to exit at a profit, and more difficult to exit with a loss, the result will be a system that exits with a profit much more often than it does with a loss. The results bear that out, and explain why our accuracy is so high.

There are some questions that come up when we consider the test though. For example, what would happen if we traded every 30 bars rather than every 20? Or, what about exiting at a different number than 2 profitable closes and 6 unprofitable ones? If this exit really does work on random numbers, we want to see it working on all different parameter sets as well.

When I first ran this test for my own edification, these were the questions that I asked myself after I realized that I was actually making money on random entries. With those results to fuel further investigation, I devised the following approach to dig a little deeper:

> *Random System*
> On every single bar, we need to simultaneously buy and sell the market. Each signal will get tracked separately, and all the results will be aggregated together.

This is an improvement over the previous test we did, because in this case we are going to cover every single possible order since we will be both buying and selling every bar. It's basically like taking *every possible trade* in the history of the ES, from the perspective of a daily system that only trades at the close. This also eliminates issues of trend, since any money made going with the trend will be cancelled out by the order placed in the opposite direction at the same time. Therefore, the only thing that will matter is how well our exits manage the trades.

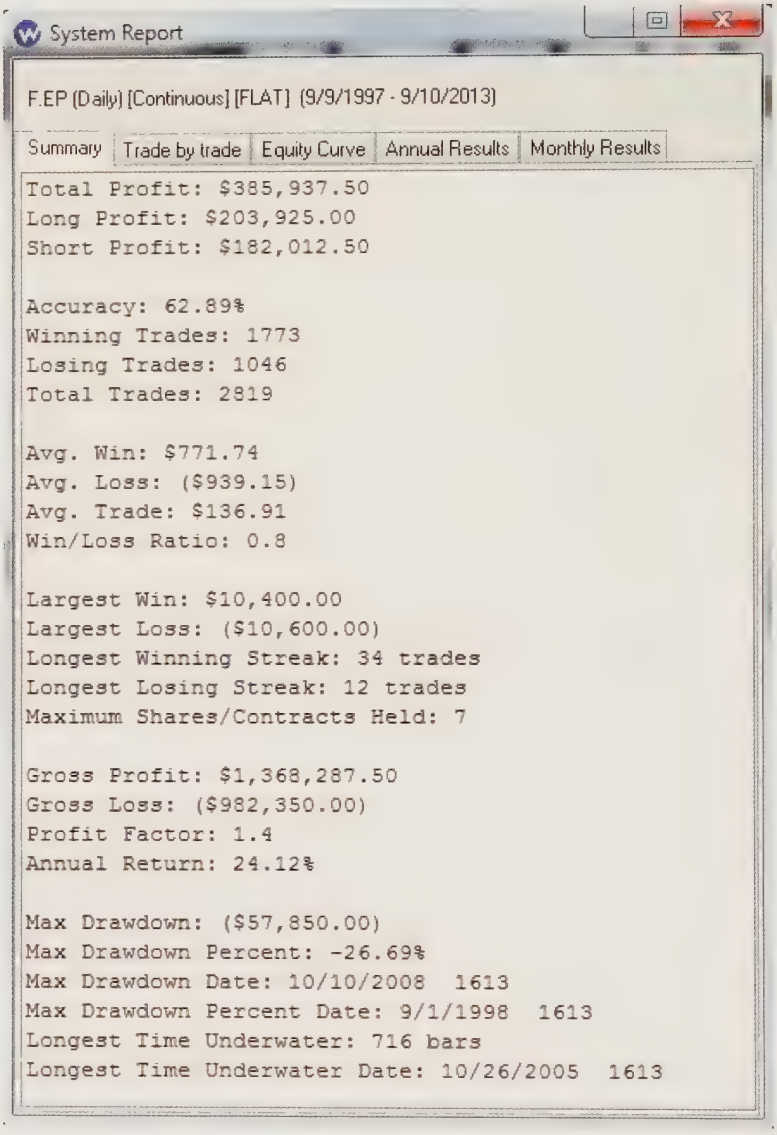


Figure 9.1: Random System with Buy/Sell on Each Bar

The results of this approach are shown above, and they are good! This system ends up trading multiple contracts, as it needs to manage a bunch of trades all at one time. At one point, there were 7 open positions running simultaneously, which gives you an idea about how it works. This is the same 2-profitable close/6-unprofitable close approach we tested earlier.

Now that we've solved the issue of when to trade (since we're trading every bar), let's take a closer look at the number of closes to hold before exiting for a profit or a loss. Again, the optimizer is the perfect tool for the job. I've run all possible combinations from 1 close up to 10 closes, for both the profit-take and the loss-take scenarios.

The Best Combinations:

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=3,#2#=10	$679,250.00	61.66%
2	#1#=2,#2#=10	$662,250.00	64.84%
3	#1#=3,#2#=9	$628,216.67	60.64%
4	#1#=4,#2#=10	$617,975.00	59.54%
5	#1#=2,#2#=9	$584,100.00	64.34%
6	#1#=4,#2#=9	$571,037.50	58.53%
7	#1#=1,#2#=10	$560,687.50	71.64%
8	#1#=3,#2#=8	$558,087.50	60.98%
9	#1#=2,#2#=8	$520,054.17	62.85%
10	#1#=1,#2#=9	$502,925.00	70.90%
11	#1#=2,#2#=7	$491,575.00	62.73%
12	#1#=4,#2#=8	$490,012.50	56.89%
13	#1#=3,#2#=7	$476,347.50	58.29%
14	#1#=1,#2#=8	$449,712.50	70.03%
15	#1#=5,#2#=10	$429,112.50	58.49%
16	#1#=1,#2#=7	$422,437.50	68.69%
17	#1#=4,#2#=7	$415,725.00	55.51%
18	#1#=2,#2#=6	$385,331.25	60.65%
19	#1#=1,#2#=6	$372,675.00	67.56%
20	#1#=5,#2#=9	$362,312.50	57.16%

Figure 9.2: Random System Buy/Sell on Each Bar Exit Optimization - Best

Above, you can see a list of the top 20 combinations. The first parameter (#1#) is the profit take number, and the second (#2#) is the loss take number. The most profitable solution was to take profits after 3 profitable closes or 10 unprofitable ones. If you look at all the others, you'll see that they are very similar in that they take profits quickly and losses slowly. This is the same premise that William Tell was based upon, only with a slightly different implementation.

The Worst Combinations:

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=10,#2#=3	($679,250.00)	37.56%
2	#1#=10,#2#=2	($662,250.00)	34.45%
3	#1#=9,#2#=3	($628,216.67)	38.62%
4	#1#=10,#2#=4	($617,975.00)	39.38%
5	#1#=9,#2#=2	($584,100.00)	34.87%
6	#1#=9,#2#=4	($571,037.50)	40.55%
7	#1#=10,#2#=1	($560,687.50)	27.54%
8	#1#=8,#2#=3	($558,087.50)	38.14%
9	#1#=8,#2#=2	($520,054.17)	36.09%
10	#1#=9,#2#=1	($502,925.00)	28.09%
11	#1#=7,#2#=2	($491,575.00)	36.24%
12	#1#=8,#2#=4	($490,012.50)	42.26%
13	#1#=7,#2#=3	($476,347.50)	40.73%
14	#1#=8,#2#=1	($449,712.50)	29.10%
15	#1#=10,#2#=5	($429,112.50)	40.67%
16	#1#=7,#2#=1	($422,437.50)	30.35%
17	#1#=7,#2#=4	($415,725.00)	43.15%
18	#1#=6,#2#=2	($385,331.25)	38.34%
19	#1#=6,#2#=1	($372,675.00)	31.39%
20	#1#=9,#2#=5	($362,312.50)	42.24%

Figure 9.3: Random System Buy/Sell on Each Bar Exit Optimization - Worst

If we look at the worst 20 combinations, with the least profitable solution at the top, we see the opposite pattern. We lose money when we take profits slowly and losses quickly. The worst case was taking profits after 10 profitable bars, and losses after only 3. No, I'm not trying to trick you here – these numbers are actually the exact inverse of the previous report, even though they were tested separately. Math has a funny way of working when you least expect it, and every solution in this list is just doing the opposite from the previous one.

So if we want to make money trading randomness, it's best to set the system up to take profits quickly when we have them, and take losses slowly. Makes sense when you think

about it – we want our systems to be successful, so we need to set them up so it's easier for them to win than to lose.

Figure 9.4: Random System Buy/Sell on Each Bar - Best Parameter Set Equity Curve

Here's the equity curve from the top solution. Very nice..! This approach only had one losing year (2010), and has consistently produced profitable returns since the beginning of trading in this market. Regarding the age old question about whether or not markets are a random walk, we can now definitively answer that with "Who cares?", because we can make money just fine in either case.

Digging a little deeper into the numbers gives us more information about how these exits work. The system report for the last run is shown below.

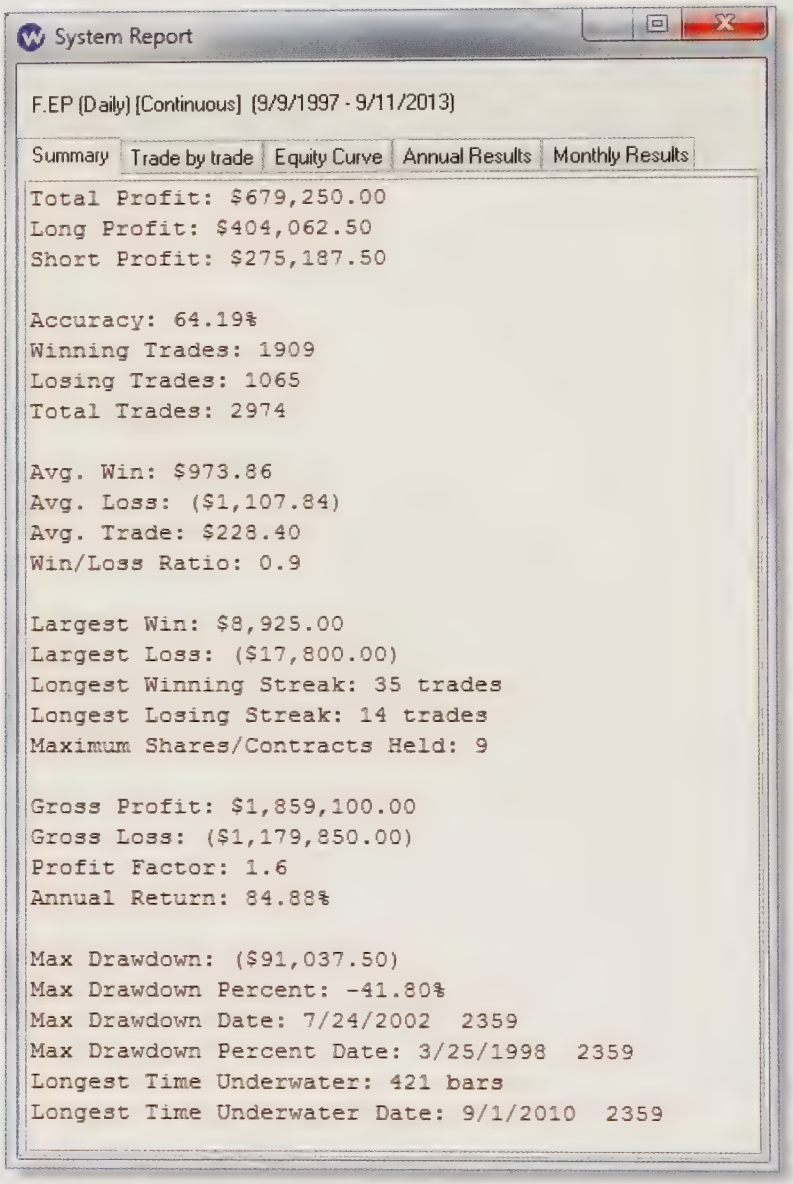


Figure 9.5: Random System Buy/Sell on Each Bar - Best Parameter Set System Report

The largest position we ever had on at one time was 9 contracts. Remember that we are holding a losing trade until it closes 10 times against us, so this would represent the worst case scenario when the market moved directionally and the system got stuck holding the wrong way almost every bar. That likely contributed to the -$91k

drawdown, which seems high at first, but really isn't considering that there was a lot more being traded than just one contract. We've got great accuracy, especially in light of the nearly even win/loss ratio, and our average trade of $228 is large enough to absorb all the costs and errors we can throw at it. There's no doubt about it, this exit makes money.

We can go one step further, and turn this into its own trading system. It's a simple transition from the previous step, and we just need to convert the last system into an indicator. This is shown in the chart below.

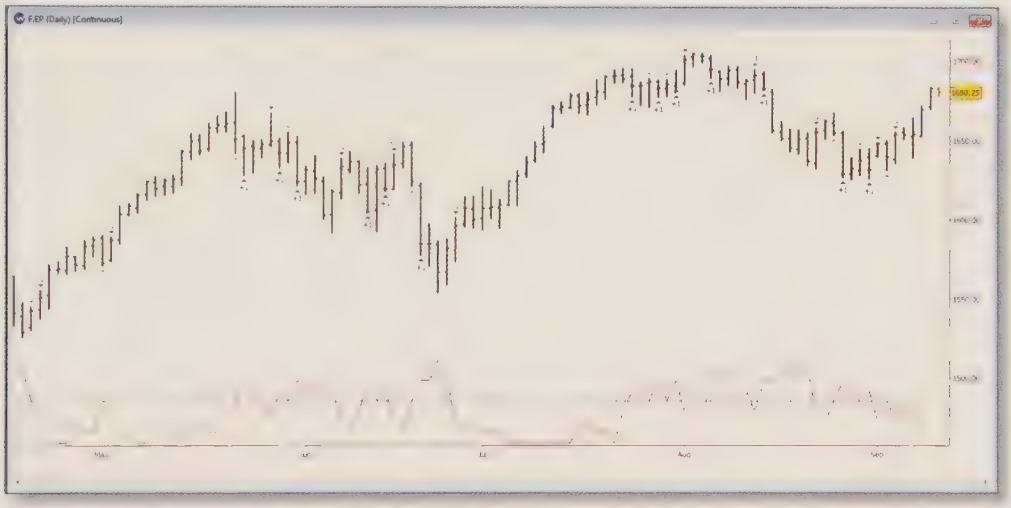


Figure 9.6: Random System Buy/Sell on Each Bar - Defined as an Indicator

The red line at the bottom is a visual look at the system we just tested. The red line tells us how many contracts long or short we are on the current bar, with the horizontal line being a flat position. Again, this is based on placing both a buy and sell trade on every bar, and running the whole thing forward using the Nth-profitable/unprofitable close exit method.

You can see that the red line tends to peak at low points, and bottom at high points in the chart. Peaks represent situations where we have the largest position of long contracts on, and troughs represent situations where we have the largest position of short contracts on. It looks a lot like one of the myriad technical indicators available to

us as traders, with a huge difference: it will backtest profitably, based on nothing but zero-line crossovers.

We've already looked at following it exactly, allowing multiple long signals or multiple short signals. Now let's look at it just trading one contract, as if it were a technical indicator. We'll simply buy when the line is in positive territory, sell when in negative territory, and get flat when it is at zero. We know that we want to make it easier to exit profitably than unprofitably, so let's run another optimization and test a range of parameters.

Optimization Results

	Parameters	Profit/Loss	Accuracy
1	#1#=3,#2#=10	$152,350.00	70.35%
2	#1#=3,#2#=9	$150,187.50	68.74%
3	#1#=4,#2#=7	$144,537.50	61.53%
4	#1#=4,#2#=10	$137,337.50	68.46%
5	#1#=2,#2#=10	$136,962.50	68.80%
6	#1#=4,#2#=9	$134,587.50	66.52%
7	#1#=4,#2#=8	$131,925.00	65.19%
8	#1#=4,#2#=11	$131,537.50	67.49%
9	#1#=1,#2#=5	$128,387.50	69.03%
10	#1#=4,#2#=6	$128,262.50	58.36%
11	#1#=3,#2#=8	$127,050.00	66.48%
12	#1#=1,#2#=10	$126,250.00	72.94%
13	#1#=3,#2#=11	$123,725.00	69.16%
14	#1#=2,#2#=11	$120,025.00	68.92%
15	#1#=1,#2#=12	$119,762.50	72.71%
16	#1#=3,#2#=12	$117,850.00	69.11%
17	#1#=2,#2#=12	$114,975.00	69.08%
18	#1#=3,#2#=7	$114,725.00	66.04%
19	#1#=1,#2#=6	$114,525.00	69.59%
20	#1#=2,#2#=9	$112,350.00	67.75%

Figure 9.7: Random System Buy/Sell Indicator - Exit Optimization

I ran the test to exit profitably from 1 to 4 bars, and unprofitably from 5 to 15 bars. The results are very similar to what we saw before, with the 3/10 combination being the

most profitable. All parameter sets produced profits, and there wasn't a losing combination in the bunch. Profitable *and* robust, just as we like it!

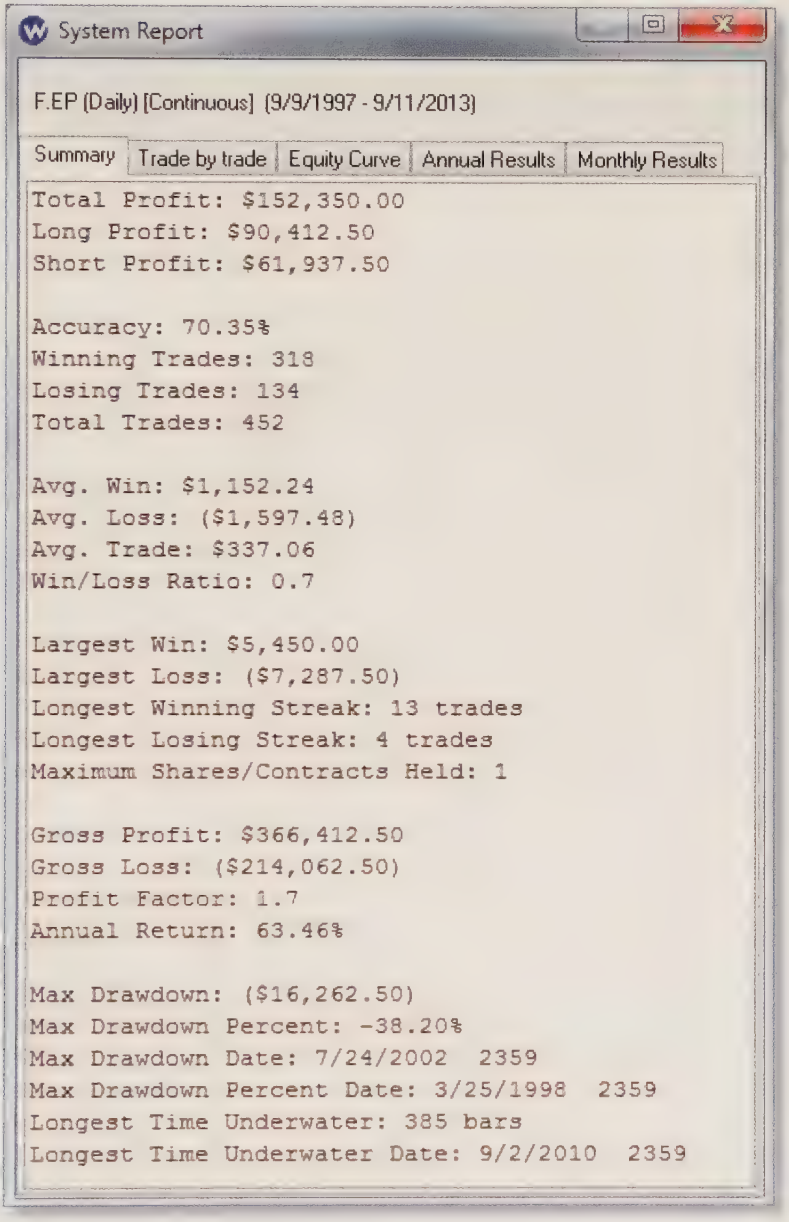


Figure 9.8: Random System Buy/Sell Indicator System Report

Check out these system results, especially in light of the fact that we aren't even using an entry signal. We've got a great profit number, fantastic accuracy, a really high average trade, and a reasonable drawdown.

This brings into question the rationale behind the attachment we all have to entry signals. Yes, the trading system vendors will tell you that entry signals are important (and that their particular signals are the best), but with results like these, how important are they really? Trading profits actually have nothing to do with getting an entry relative to some important point on the chart itself. It is all about the relationship between where you entered and where you exited. So if you exit at a better price than where you entered, you make money. If you exit at a worse price, you lose money – and that's all there is to it. There are actually no extra points earned for selling the high tick, or buying at the bear market low, or calling them out ahead of time in a public display of forecasting prowess. A dollar is a dollar, whether it is made at the bottom of a move, in the middle, or at the very end.

Trading is more about getting in *somewhere*, and then managing the trade to get you out at a better price. If you look at it that way, you'll see that the entry itself is almost meaningless, which is why we are able to build systems like this. The search for trying to enter at tops and bottoms on the chart itself is just another piece of brainwashing that we pick up as we learn this craft. What does that say about the hard core WD Gann followers who spend 20 years trying to figure out how to detect those tops and bottoms ahead of time..?

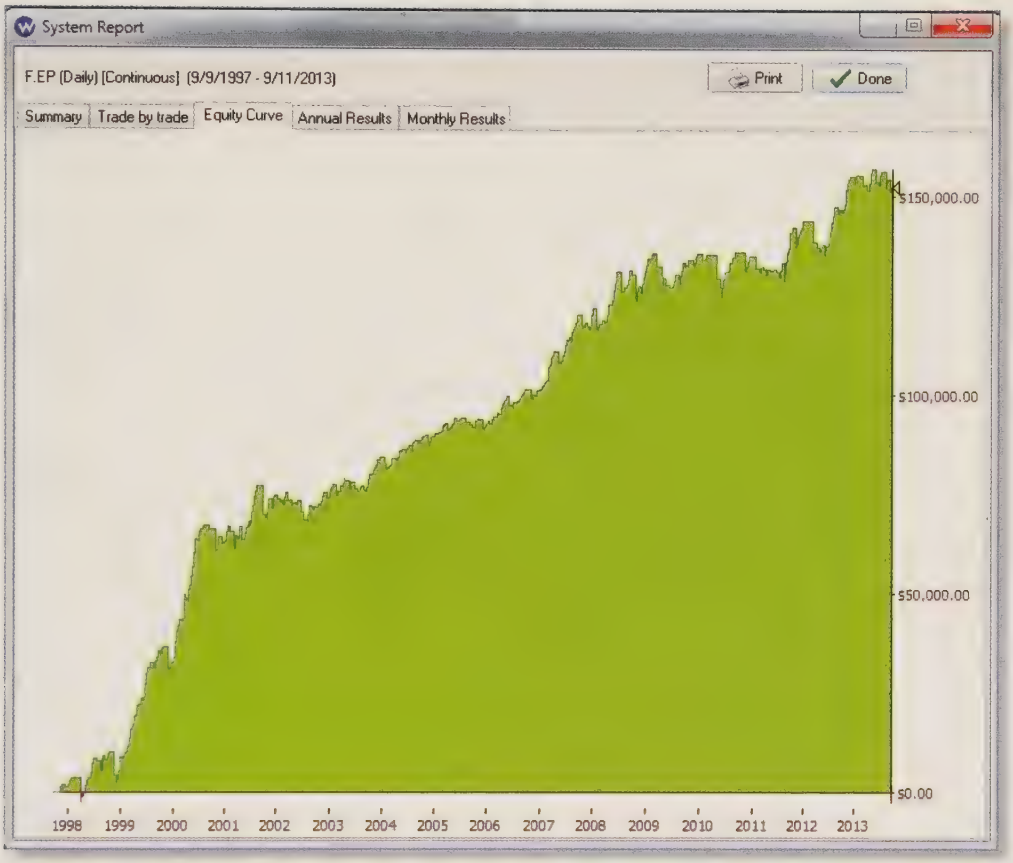


Figure 9.9: Random System Buy/Sell Indicator Equity Curve

The equity curve is nearly identical to the one we just looked at in Figure 9.4, but I want to include it for completeness' sake. It pretty much speaks for itself.

So what does all this mean? It means we've now got a way to profitably trade markets, and 100% of the edge comes from our exit method. Because of that, we are free to merge this with whatever sort of analysis method strikes our fancy. If we are technically inclined, we can find setups based on squiggly line indicators and complicated mathematical formulas. If we like to draw on our charts, we can pull out our Gann Fans and trendlines. If we're more esoteric, open up that ephemeris and figure out those planetary aspects. Heck, we don't even need to bother to backtest any of those approaches! This will work even if we flip coins or guess, so it should work just fine with any of the more refined random number generators available to modern traders. And if

we can pair it with something that actually does provide an edge on the entry side, we'll be able to grow the account even faster.

Hopefully, this chapter opened your eyes a bit, and dispelled one of the most pervasive myths in the trading industry. Entry signals are definitely fun to play with, but they don't have as much to do with profits as we would initially think. In fact, we can build systems without needing any entry signal at all, and we will still make money over time. We find ourselves in a similar situation as we did in Chapter 6, when we discovered that stops and targets actually don't add anything to our trading method. What do we do if adding entry signals to our random system can't actually improve the results...? You already know the answer to that question.

10. Merged Systems

At this point, we've looked at quite a few different systems with some very appealing performance statistics. So what is the next step? Pick one that we like and start trading it? Although that is definitely a possible option, we aren't quite done, and there are ways to hammer our trading edge down even more firmly.

Throughout this book, I've tried to stress the importance of being aware of the fact that parameters like to move around, and that markets in the future don't behave quite like markets in the past. Everything that I've said with regards to parameters and the care you have to take with them also applies to individual trading systems.

A trading system can really be thought of as one big parameter that has been tuned to a specific data set. If we have done our work properly as developers, we have created systems that will be resistant to the market's tendencies to move around on us, but being resistant to changing market character is different than being immune to it. Luckily for us, we have already seen the solution to this issue, and have been working with it since chapter four.

If we can consider our systems as parameters, what do we do to help shore them up against decay? First, we check to make sure that most of our parameters make money, so that we are trading a profitable idea rather than a profitable parameter set. Next, we try to find a way to trade multiple parameters all at once, as with the case of Hive Technology, so that our results aren't dependent on any one specific parameter setting. In other words, we leverage the Law of Averages so that we can surf across a wide range of profitable parameters in the unseen data set where real money is actually at risk.

In our case, we've got a range of systems that we've gone through, each of which makes its own case for being allowed to manage money in real time. What we are trying to avoid is the situation where we select the system we like best, which turns out to be the dud among the group, and we end up spending the next few years in the extremely uncomfortable position of trading a system that has found itself in meltdown mode. So rather than trade one of them, let's diversify and trade all of them at once. That way, even if one or two of them end up moving into a drawdown, the others will be there to counter balance things on the profitable side.

What we are talking about here is diversification, but in a slightly different way than the context in which that term is usually used. If you are a Finance professor, and don't know how to actually trade markets, you are going to spend a lot of time figuring out how to allocate money in the right way as an alternative. This is why diversification plays such an important role in Modern Portfolio Theory.

From a systems perspective, we are still looking at just one market, but we are going to trade it using multiple systems. So we are going to diversify across systems, rather than across asset classes. Think about it as trying to keep our balance on a moving platform. If we are standing on one leg, it's very easy for us to lose our balance and fall down. If we stand on two legs, we have a better chance of staying up, but if we get down on all fours, it's going to be really difficult to knock us over. One of our legs can get knocked out from under us, but the other three points of contact will keep us in position. Similarly, if we spread our trading across multiple systems, one of them can blow up, but the aggregate equity curve will still be OK.

When diversifying across systems, it's important that the individual systems aren't all based on the same thing. There really isn't diversification if we trade 10 systems all based on moving averages. In that case, we're really trading one method (averages), just implemented differently. What we really want is for signals to be coming from as many unrelated areas as possible. So if one system is based on averages, let's use another one based on momentum, and even better, another one based on fundamentals. It's all about getting as wide a stance as we can for stability.

From the perspective of this book, we've looked at a few markets, and lots of different systems. Let's pull most of them together into a single market and trade the E-mini S&P 500. Not only is it a huge, very volatile market, which is good for us as there are lots of opportunities, but most of what we've looked at in this book will work on it.

Here's a brief listing of what we've got that we can use to start trading the ES:

System Inventory	
System	**Based on**
BCI	Mean Reversion
Lunar System	Spherical Astrology
Chicago Hive2	Hive Technology, AMAs
Inverse Momentum	UltraSmooth Momentum, length = 2
GA1	USM8, plus bar filters
GA2	USM2, USM10, USM18 blend
Random Exit	Random Numbers

Table 10.1: List of Systems Available for Trading

We have seven systems we can work with to trade the ES. Three of the systems (Inverse Momentum, GA1, and GA2) are based on UltraSmooth momentum, so we're a little biased towards momentum, but at least those systems use four different USM parameter lengths, which helps. All of the rest are based on unique concepts, including Mean Reversion, Spherical Astrology, Hives using adaptive averages, plus our random system from the last chapter. I particularly like that we've got an astro system, a machine learning algorithm, and a random system involved, as otherwise we'd be concentrated completely in the technical indicator realm. So we're not 100% diverse, but we've got enough diversity here to give us a good base.

You'll notice that William Tell is missing from the list. That system was specifically created for Bonds, and doesn't work very well on the ES. We want all of our base systems to be robust systems worth trading independently. If a system can't give you that much, don't include it in your basket. Although we only briefly talked about it at the time, notice that William Tell is actually its own amalgamation of various patterns, which is one of the reasons why it worked so well for so long. As we saw in Chapter 2, more than one of those William Tell patterns self-destructed shortly after the time when the system went live, yet the system still had a long lifespan of around 10 years in real markets. That's because all the other patterns contributed to hold the overall results up despite the few rotten apples. That's exactly the reason combining methods is such a great idea.

Coming back to the merged ES system, there are two ways to do this. The first way is to go forward and actually trade every system at the same time. When following an approach like that, systems will often cancel each other out. If one system places a buy signal, and the other places a sell signal, then at the portfolio level you're really just flat,

as the profits made on one system cancel the losses taken on the other. So in that case, you don't actually place trades from both in separate accounts, but rather just stay flat and save the commissions.

However, different systems trade in different ways, so you have to make sure to account for each system and follow it properly. The BCI system is going to be trading at the open, rather than the close, so you'll need to make sure to enter and exit it appropriately. The Lunar System is not always going to be in the market, and will be taking profits off at the Nth-profitable close using its own exit method. All the other systems would trade right at the close itself. So practically speaking, you'll need to be able to place orders both at the open and the close, and will have to adjust your exposure to match whatever the aggregate is doing. You'll also need to be able to trade multiple contracts at once, as there will be certain times where all the systems agree with each other, which would result in being long or short a full 7 units at once.

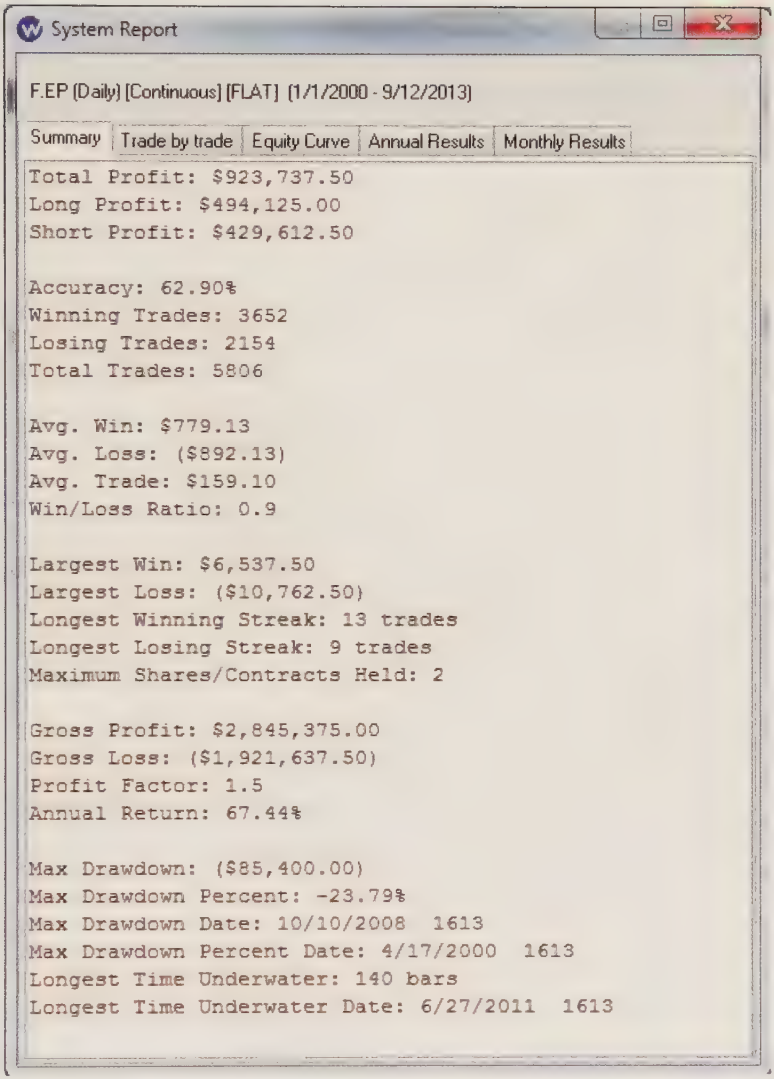


Figure 10.1: Simultaneous Merged System Report

The report above is the result of all 7 systems being traded together in the same account, independently of one another. This approach generated huge profits, but also has a large drawdown. Everything gets multiplied over since we have to trade multiple contracts in this situation. The equity curve gives a better picture of what is going on:

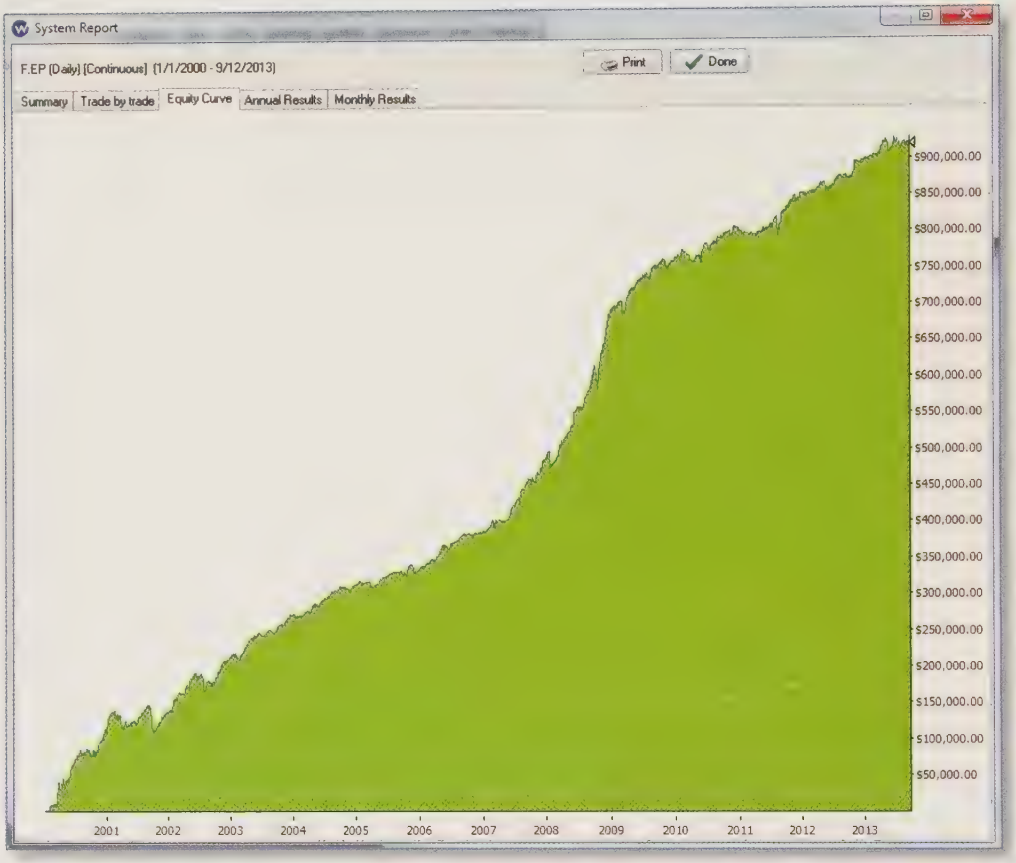


Table 10.2: Simultaneous Merged Equity Curve

That's a pretty smooth equity curve! If a picture is worth a thousand words, this one should explain the advantages of merging multiple systems better than me talking about it. The combined equity curve is superior to any of the equity curves from the component systems, and we didn't experience a single losing year.

So that's the most obvious way to combine systems. However, it can be tedious to manage, and you do have to have a large account in order to implement it. What I prefer to do is to create a brand new system that trades just one unit, but does so using a combination signal from all the individual systems.

The easiest way to accomplish this is to simply give each component system a vote, and then to let the majority rule when it comes to making decisions about whether to be

long or short on a given day. In order to do that properly, there are a couple requirements that must be met. Since we're only going to be trading one unit, we have to decide at what time we are going to place our orders. Most of the systems trade on the close, so that makes the most sense in our case. As we will be taking a vote at the close to determine direction, we also really need our systems to be able to vote each day, and don't want any particular system to be sitting on the sidelines. This will require a few changes to some of our components. Specifically, we're going to need to have the BCI tell us what it wants to do at the close, not the next open, and we also need for our Lunar system to give us a trading indication every single bar.

You can see why I tend to really like to create reversal systems. They lend themselves very well to the merging approach we are discussing. When each system provides a signal every single bar at the close, and is never flat, we have a great candidate for a voting environment. After implementing the changes mentioned, converting the BCI to trade at the close, and using the Lunar Reversal method rather than the Nth-profitable close method, we are ready to set up our voting approach.

There are lots of ways to do this, and we could try weighting the individual systems by system performance, or even dynamically by watching the equity curve as it unfolds, but a lot of times the best choice is just to take a simple vote, with each subsystem receiving the same amount of voting power. Given the fact that diversification is the reason we are doing this, we really don't want to give extra power to any one component. Since we don't know which systems will go into drawdown, we really don't want to let any one system overrule the others. As long as our systems really are based on different things, the odds of one system failing is significantly higher than the odds of all of them failing at once, and we need to set things up so that our account only loses money in the least likely scenario, which means equal weights is the safest approach.

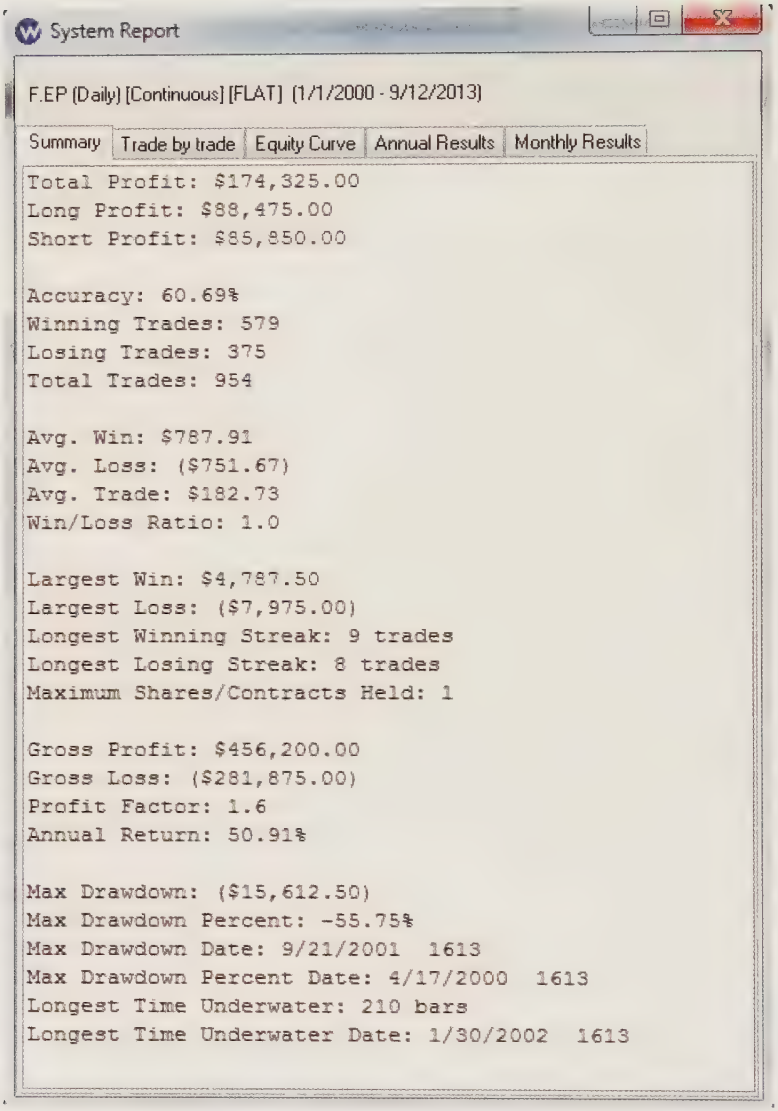


Table 10.3: Voting System Report

Here's the performance report for our voting system. Good, solid numbers. Don't pay any attention to the annual return or drawdown percentages just yet. We'll get into actual returns on account next chapter.

Table 10.4: Voting Equity Curve

The equity curve is not quite as smooth as the last one, but that is to be expected as we don't have seven units running at a time to smooth the really small bumps out. This is still an awesome equity curve.

Let's do a quick comparison, and see how the merged results stack up against all the individual components.

System	Profit	Drawdown	Profit/Drawdown	Equity Curve
BCI (trading the close)	$85,337	-$20,000	4.26	C-
Lunar Reversal	$81,937	-$25,912	3.16	B
Chicago Hive (Reversal)	$80,637	-$19,812	4.07	B-
Inverse Momentum (USM2)	$152,700	-$15,337	9.95	B+
GA1	$176,162	-$21,937	8.03	A-
GA2	$191,262	-$17,162	11.14	A
Random Exit (3/10)	$111,037	-$16,262	6.82	B+
Voting System	$174,325	-$15,612	11.16	A

Table 10.5: System Results Comparison

The table above lists each system, its total profit, max drawdown, profit/drawdown ratio, and my subjective rating of the shape of its equity curve. Since the voting system takes input from each system in this list, we aren't expecting it to be the most profitable. We do expect it to be the most stable though, and you can see that it definitely makes more money than the average system, and has practically tied Inverse Momentum for the lowest drawdown. It also has the highest profit/drawdown number, which measures the amount of profit made for a fixed unit of risk. Most importantly, it has the cleanest, straightest equity curve of the group.

You may notice that GA2 has a higher profit number, almost as low of a drawdown, and basically the same profit/drawdown number, while having almost as nice of an equity curve. Why not just trade that one system rather than the voting method? The reason is that the voting method will still work even if GA2 manages to catch fire and go down in flames. Had the GA2 method been responsible for trading the account, you'll find that the account craters right alongside the method, whereas in the voting system, all the other methods can step in to help keep the boat afloat during a similar crisis. In fact, that's exactly what happens when individual systems go into their respective drawdowns, as proven out by the stability of the equity curve. Remember, like parameters, we don't have any clue as to which systems will be the best performers in the future, and odds are that the best ones will not be what we expect, which is why we

are more concerned with stability at this point than a few extra dollars of profit. We're going to see how to squeeze quite a bit more than a few extra thousand dollars out of this system in the next chapter, but in order to do so, we need to make sure that we are trading as stable a method as we possibly can.

Combining individual systems together into a single buy/sell signal is the second way to merge systems, and has some distinct advantages over simply trading all the systems together using multiple contracts. The first advantage is that we reduce overall account risk. Although the initial merging generated $923k in profits, it did so at the cost of a max drawdown of -$85k. The profit/drawdown number in its case is 10.81. This is close to the 11.16 number as posted by our voting system, but not quite as good. That is especially important as three of the components in the multiple contract method were improved versions of what we used in the voting method. The BCI was able to trade at the next open, which resulted in significantly increased accuracy, and the Lunar System Nth-profitable close approach, as well as the Chicago Hive2 system both outperform their reversal method counterparts in multiple categories in their system reports. Despite these facts, the voting method is able to hold a slight edge against the multiple contract approach, which shows the inherent strength of this method.

The second advantage the voting method has over the multiple contract method is that it is significantly less complicated to manage, which shouldn't be overlooked. Rather than trading seven systems with seven different ways of entering and exiting, we are trading one system that simply trades at the close. This is going to be easier for us to manage, and we will make fewer mistakes. Don't underestimate the power of confusion, especially when there's money on the line and time pressure to get orders in. If you can accomplish the same task with one-seventh the amount of effort, that's a huge advantage.

You can see how all of this is now coming together. We find various ways to exploit the different edges we have in trading markets, and we then turn those methods into mechanical systems. This is done multiple times, and we eventually find ourselves with a collection of good systems. We then take all of those systems, and find a way for them to join forces together, which produces a result that is better than any of the individual systems were able to attain on their own, or even in aggregate. As Aristotle would say, the whole is greater than the sum of its parts.

As you have been making your way through this book, if you have been wondering to yourself which of the various approaches would be best used in your own trading, the answer should now be clear. You really want to use all of them together, plus whatever other ideas you've already got cooked up on your end.

11. Position Sizing

As mentioned in a previous chapter, in order to actually complete a trade in our account, four things must be known. The first thing is that we need to know what market to trade. Next, we have to know when to place our entry. After that, we have to know how many contracts or shares we are going to trade, and finally, we have to exit our trade in order to realize our profits or losses. Without any one of these four things, we aren't going to be able to trade, and all of them are very important to our overall success.

The first item, which market to trade, is largely up to the preferences of the individual trader. Some markets are arguably more suited to speculative trading than others due to liquidity and slippage issues. For example, if you are trading a thin market with no volume, you are going to have trouble executing your orders at reasonable prices, and the slippage you experience will result in reduced performance when compared to trading a very liquid market. Additionally, if a market is very flat and boring, it can be difficult to find good trading opportunities. Luckily for today's traders, we don't have to worry about volatility (at all!), and we've got plenty of markets with extremely high volumes, so market selection isn't something that needs to be an obstacle.

Entries and exits tend to get the bulk of the focus, especially among beginning traders, because those are the most interesting aspects of trading. The entries and exits tell you when you are going to trade, and are responsible for whether or not the trade is a profitable one. Most of this book so far has been all about managing either the entry, or the exit, or both in order to produce profitable returns over time.

Position sizing handles the last piece of the trading puzzle, which is how large a position to put on during a given trade. For some reason, position sizing tends to be looked upon as a strange side topic, not nearly as interesting as entries and exits, and gets basically glossed over in most of the trading literature. If you quickly browse through the inventory in any of the online trading bookstores, and compare the number of titles related to entry signals compared to the number of titles related to position sizing, it will be obvious how little attention gets placed on this aspect of trading.

The funny thing is that all traders must practice position sizing, because in order to place a trade, a trader must specify the size of the order. So whether or not a trader

understands or appreciates position sizing, they will have adopted a position sizing method in their own trading by default, simply by placing a trade through their broker. For example, "I'm going to be conservative and trade one contract" is actually a position sizing approach all on its own.

Although entries and exits are responsible for whether or not a system generates a profit, position sizing largely governs how large that profit will be when looked at over a period of time. For example, in the last chapter, we took a look at a voting system that generated $174k over the last 12.5 years trading the Emini. If we had started trading that system back in 2000, we would actually have that amount of money in our account today. But if we had traded two contracts instead of only one, we would have $348k in our account instead of only $174k, twice the amount in dollar terms. All the signals were identical, the amount of work to implement them was identical, but in one case we made twice as much in dollar terms as in the other. That's the difference that position sizing makes.

If you are interested in growing your account out to eventually earn millions of dollars per year, that is a completely reasonable goal. That possibility is one of the reasons why so many people get enamored with the idea of becoming professional traders. No trading system on its own will be able to generate millions in profits though - it is the position sizing that will do that for you. If you just want to trade as a hobby, and make a little here and a little there, you don't need to pay too much attention to position sizing. But if you are really taking this seriously, and want your trading to have maximum impact on your life in the next 5-10 years, you really need to understand and implement position sizing properly in your trading operations.

There are lots of ways to do position sizing, and like anything else, some methods are better than others. The most obvious position sizing algorithm is one where we trade one contract for every $X in our account. This is where the one-contract system reports help us out. Let's pull the report back up from the voting system we built last chapter:

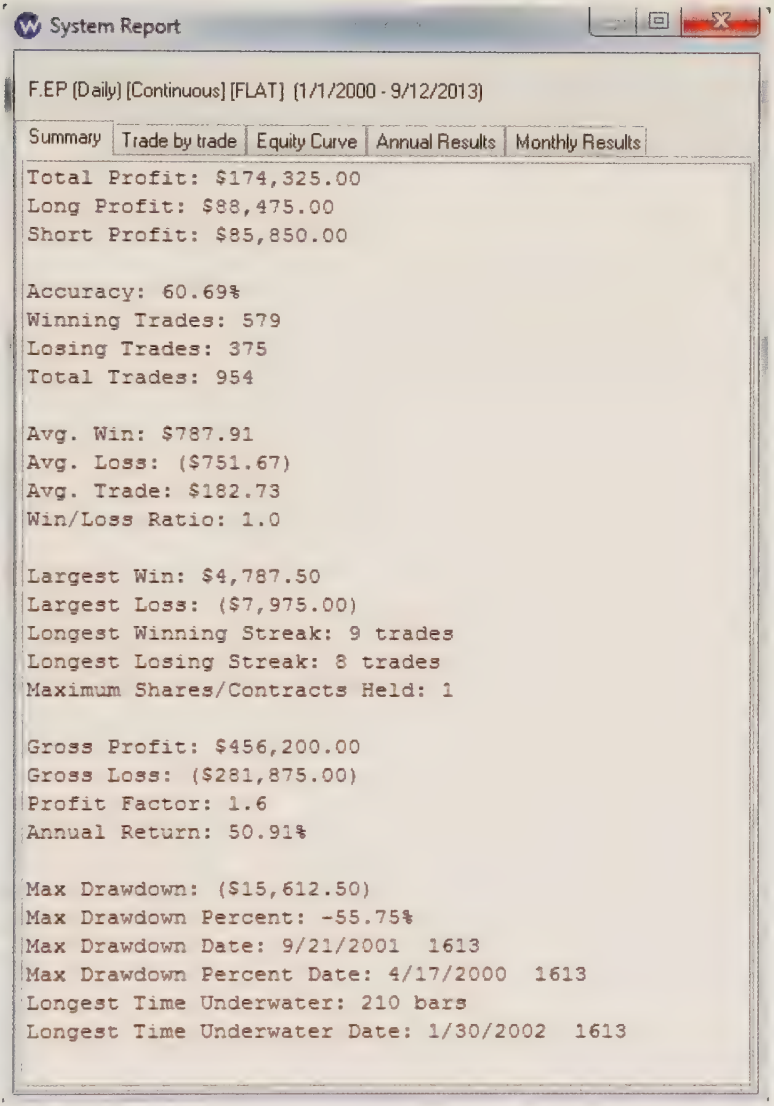


Table 11.1: Voting System Report

What we need to do is figure out what a reasonable account size is in order to trade one contract. This is best done by considering the worst case scenario, to determine how much we'd need in order to trade just one contract assuming we were about to hit a significant drawdown period. In the case of this system, our worst drawdown ever was a loss of $15,612. The margin needed in order to trade one Emini is around $5000, so we'd have to have at least $20,612 in our account in order to be able to survive that

drawdown and be able to keep trading. That would represent the most aggressive we could really be with this method. Technically, you could trade with less than that, but if you experience a drawdown early (and you will!), you'll basically lose the entire account as a result of being undercapitalized.

We need to plan for success, and be ready to weather those storms, so I use the method described back in Chapter 1, and determine my "uncle" point in percentage terms, and simply divide the drawdown number by that. As I mentioned, I'm comfortable accepting enough trading risk to allow for a 40% drawdown to my account, so in this case, I would divide $15,612 by 0.40 to arrive at $39,030 as the amount needed in order to trade one contract.

What that really means, according to this particular position sizing approach, is that I am allowed to trade one contract for every $39,030 I have in my account. If I have a trading account of $100,000, that means I can trade 2.56 contracts. We can't trade fractional contracts, so we round down since we're conservative to 2.

This may seem like a pretty big account in order to just trade a couple of contracts, but I'm doing the math as if this were a business. A lot of traders are way overleveraged, which is why so many of them can't weather drawdowns. The only way to trade this system with less than $39,030 in the account is to increase the acceptable percentage risk number, or hope that a drawdown doesn't happen. If you are hoping, you are also requiring that luck be on your side, which means now you're gambling. I've found that whenever I give them the chance, markets tend to like to make withdrawals from my account rather than deposits. Because of that, I don't gamble when it comes to markets. If I didn't have enough to trade one Emini contract, I'd go trade the SPY using 10-share lots instead. Rogue waves aren't as uncommon in markets as one would think, and getting an account blown out isn't something you really want to happen to you if you can help it.

Let's now start trading with our $100,000 account back in 2000, using the same system that we've been working with. As our account grows or shrinks due to trading profits and losses, we will adjust the size of our trades so that we are always trading one contract for every $39,030 in our account. This allows us to compound our profits into larger size, but also reduce our size when drawdown periods strike.

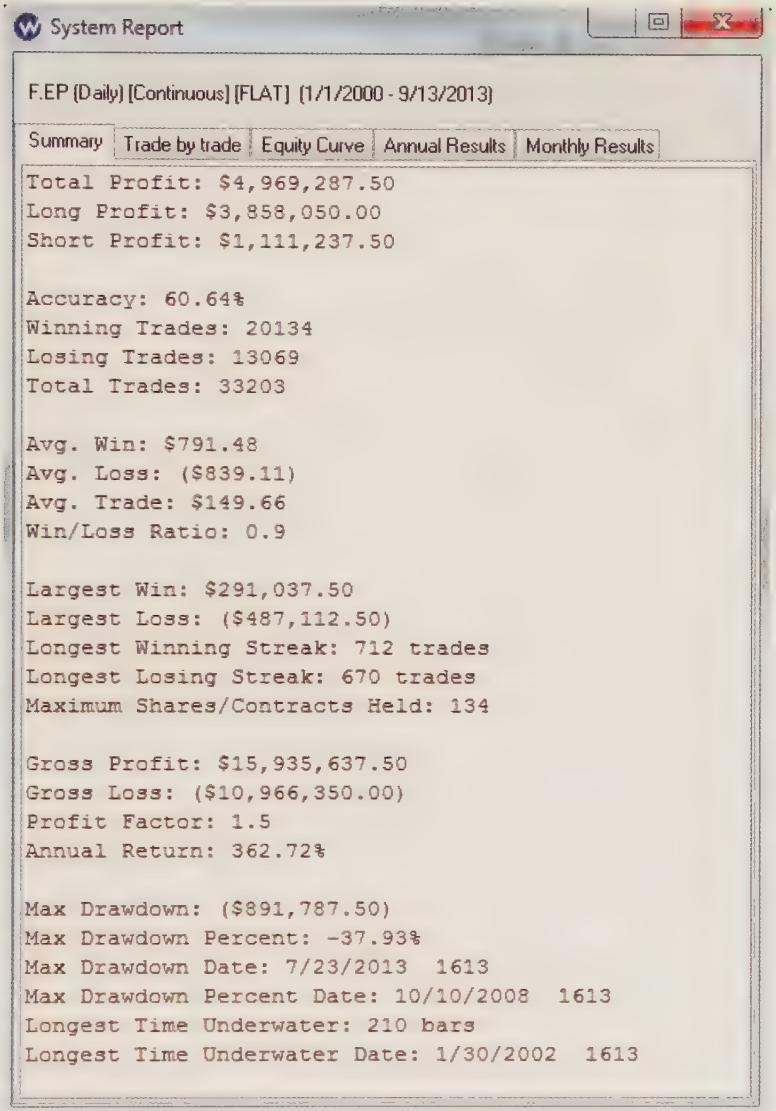


Table 11.2: Voting System with Position Sizing System Report

You can see that our system generated almost $5M over this period, which is significantly more than before. We had almost a $900k drawdown at one point, but remember that the account was much larger then. As a percentage, it was right around a -38% drawdown, just like we had planned.

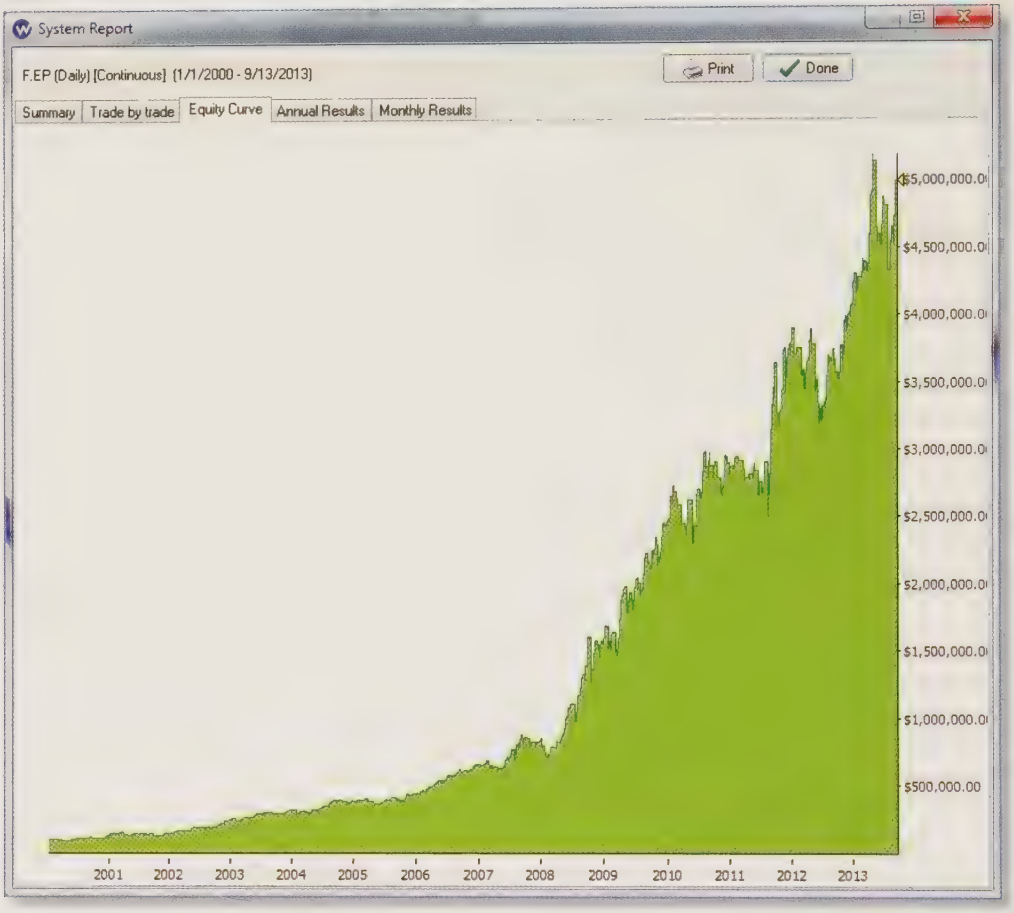


Table 11.3: Voting System with Position Sizing Equity Curve

The equity curve looks a little different than it did before. When we apply position sizing to a system, a straight equity curve will end up getting transformed into a curved one. That's because, from a dollar perspective, the results end up becoming exponential instead of linear. You can see where this happens on the curve right around 2008. Things were relatively nice and straight up until then, and then the curve explodes. You want to get to that point as soon as possible, as that's where the real money is made.

So what sort of annual percentage gain is our system giving us? Although our system report has a number there, that number refers to a 1-contract method, where profits are taken out each year. That's not exactly the same as what we are doing anymore, so we need to figure this out ourselves.

The formula is:

$$Annual\ Return = \sqrt[Num.Years]{\frac{Ending\ Account\ Value}{Beginning\ Account\ Value}} - 1$$

Plugging our numbers in, we obtain the following result:

$$Annual\ Return = \sqrt[12.5]{\frac{4{,}969{,}287}{100{,}000}} - 1 = 36.7\%$$

So our system has generated a 36.7% gain every year for the last 12.5 years. Not bad. There aren't many legitimate investments available that can provide that sort of annual return, and that includes most of the hedge funds in existence.

Let's run this system again, but using a different starting value. Let's say we've got $1M to put into this system right from the start, and do everything else identically. In that case, we end up generating over $59M by the time 2013 rolls around, for a 38.6% annual return. Notice that although our starting value was 10x larger than before, our ending balance was more than 10x larger, and we gained two percentage points on our annual return. The reason for this is that larger account values don't suffer from rounding errors on the contract sizes as much as smaller ones do, which happens a lot at the beginning. Money makes money, and you can see it with these simulations.

It should be obvious how someone using active position sizing would seriously outperform someone else just blindly trading a flat number of contracts. As profits are made or lost, the effects are immediately implemented in the next trade taken, which allows winning streaks to gain massive traction in the account.

This is the default position sizing approach that most traders tend to employ, and it works pretty well. It is definitely better than not having any position sizing strategy! But there are other methods as well, and I'd like to share a variant that I like to use in my own trading.

I do like the basic idea of trading a certain number of contracts for a fixed dollar amount. The approach makes intuitive sense to me, and it can be used to run an account up very quickly if the underlying system is a good one. It does have a weakness

though, and the weakness is that the largest positions always happen on losing trades. After trading this method for a while, eventually the account will be large enough that it adds or reduces contracts every single trade. This occurs when trades make or lose more than the $39,030 value required in order to trade one more contract in the account. As the equity curve is rising, each trade will add more contracts to the working orders, and we will be increasing our size each time. Eventually, we'll hit a losing trade, and we will reduce our size. The number of contracts placed during that losing trade will be larger than the contracts placed on any of the winning trades leading up to it. Because of this, our position sizing algorithm has the undesirable benefit of over-emphasizing losing trades, which is not something that we really want to have happen.

My solution to this problem was to modify the approach so that it works in terms of percentages of the account rather than on fixed dollar values per contract. In this modified approach, like before we first determine how many contracts are traded per dollar in the account. This calculation is done once at the beginning. We then start trading this amount, and continue trading it until either the account gains X% of its value, or loses Y% of its value. If the account doesn't reach one of these thresholds, we maintain our current position size. We only get to change our position size when the account size penetrates our upper or lower thresholds. When it does, we use the current account balance to solve for a new trading size, as well as new threshold levels, and the process repeats.

When trading this way, we want to make it easy to increase our position size, but more difficult to decrease it. This is very similar to our random entry system, where we make money by making it easy to take a win, but harder to take a loss. I usually set the thresholds so that position size is recalculated when the account gains 0.5% of its value, or when it loses 30%.

The effect of this change is that if we make a marginal new high in equity that allows us to increase our position size, but then we take a loss right there, we won't by default have to immediately reduce our position size. We only reduce the position size when in a full-on drawdown situation, because the account must lose a substantial part of its value in order to reduce its size. Conversely, we are always ratcheting up the size as we win. So it's easy to get big, and more difficult to get small.

Given the fact that we have accounted for drawdowns when computing the minimum dollar value to trade one account, we are already protected against losses at the portfolio level. Whether the drawdown happens with one contract or fifty, as a percentage it will still remain within our expected range. So as long as our account is large enough before the drawdown, we should be able to weather it without having to resort to much reduction in our position size. If that reduction does happen, it only happens during the worst drawdown periods, and then we make it easy for the account to gain size on the rebound.

Let's apply this modified method to the system, using a starting account value of $100,000 again:

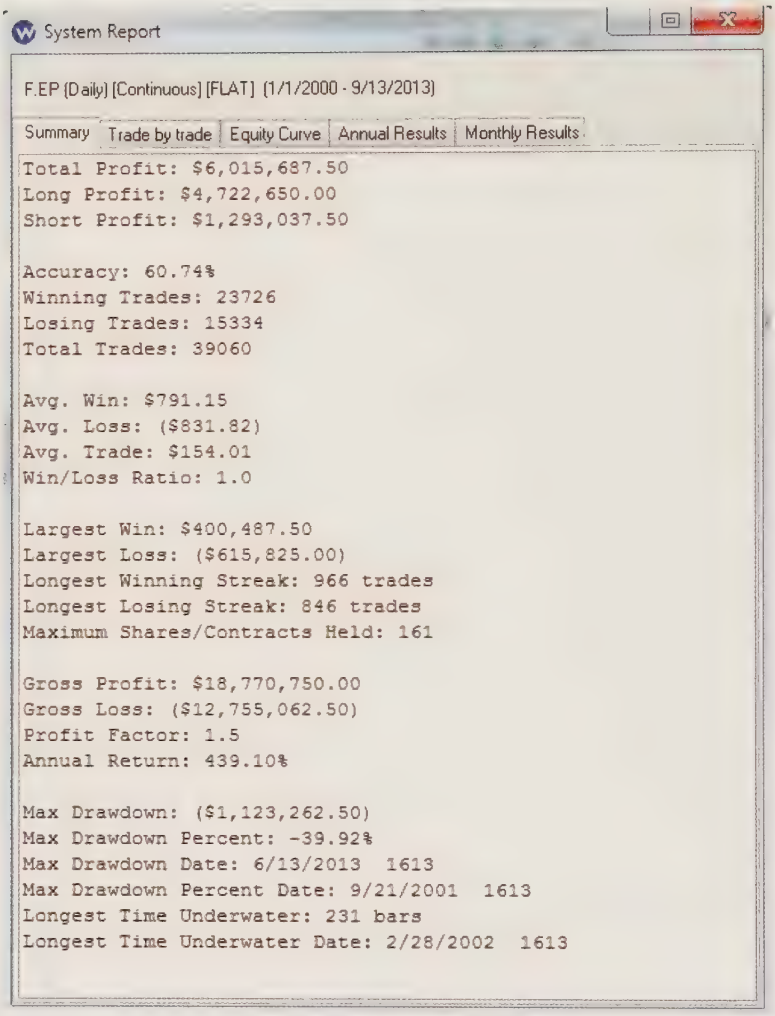


Table 11.4: Voting System with Enhanced Position Sizing System Report

The modified approach earned $6M over this period, 20% more profit than the original position sizing algorithm, while keeping our drawdown almost the same as a percentage. It has upped our annual return to 38.7%.

You can see how profitable futures trading can be, once you start paying attention to all the variables that you have control over. There isn't a single-contract trading system in existence that can beat the profitability numbers generated through correct application of position sizing algorithms, yet most traders tend to never get past that one-contract

trading mentality. It's definitely an eye-opener once you do a couple of runs like this on your own. I highly recommend it.

Now that we've seen the power of position sizing, it's time to come back again to diversification. We've seen how important diversification is in terms of individual parameter solutions, and how by trading multiple parameters at once, we can improve stability and mitigate issues of parameter migration. We've also seen how merging many systems together into a voting method increases performance to a new level unattainable by any of the component systems. We can do the same thing with position sizing.

Position sizing algorithms also have parameters to them, and those parameters tend to govern how aggressive the algorithm is in increasing its size over time. If you had to trade your entire account using a very aggressive setting, you probably wouldn't be able to keep it up very long, as the stress would be overwhelming. There aren't many people that can stand to watch the majority of their account evaporate in a drawdown and manage to keep trading the system despite the fact. But there's also no rule that says we need to trade just one system in our account, or even just one account.

If we went into an appointment with our wealth manager, they would sit down and lay out a number of investment vehicles for us. Each vehicle would ideally be uncorrelated from the others, and our manager would recommend that we put some portion of our assets in stocks, some in bonds, some in various funds, etc. Additionally, our manager would have broken our investment vehicles down by how risky they were. We will put most of our assets into safer vehicles, but in order to maximize overall portfolio return, we also would be considering putting some small portion of our account into riskier assets, such as hedge funds or managed futures. There's always a chance that we get clobbered in those investments, but the potential return is high enough that we are willing to take the risk, and we have enough of our portfolio in other areas that we'll still live to fight another day in that situation. We can approach our own trading in a similar fashion.

The way to do this is to set up multiple accounts, each trading a different position sizing approach. One of the accounts is the aggressive account, and it has one purpose, to swing for the fences and try to hit a home run. You aren't going to put all your eggs in this one basket, but given the potential returns we can generate with an aggressive

position, it makes a lot of sense to at least let one of the systems swing for the fences. Yes, you will likely see some nasty drawdowns, but you'll only see it in one account, and the other more conservative accounts will hold the bulk of your trading, which will be done in a way that is much easier to watch.

Here's how I'd manage the aggressive version of this system. We know that our drawdown is $15,612, and that the margin to trade the ES is $5k. That gives us the minimum required value to trade one contract at $20,337. If we hit exactly the same worst drawdown as we did in the past, we would be able to just squeak by and continue trading with this size account. Let's now add a small pad to account to account for the fact that the worst case drawdown probably hasn't been seen yet. 50% is a good amount to use, and if we trust ourselves and our methods we can use a smaller pad, like 25%. If we go with 25%, that gives us $15,612 max drawdown + $5000 margin + $3903 pad = $24,515. This then, is the value we use when trading the extreme version of our method.

After we have calculated this number, we then set up a separate account for our aggressive system, give it our blessings, and tell it to come back with its shield or on it. The purpose of this system is not to grow a retirement account. The purpose of this system is to give you a chance to take home the jackpot by leveraging your method to the hilt.

If the system blows up, it blows up, and you'll have to do things the slower way using your other accounts. But if it works, it will REALLY work. Here is how our merged system performed using aggressive position sizing:

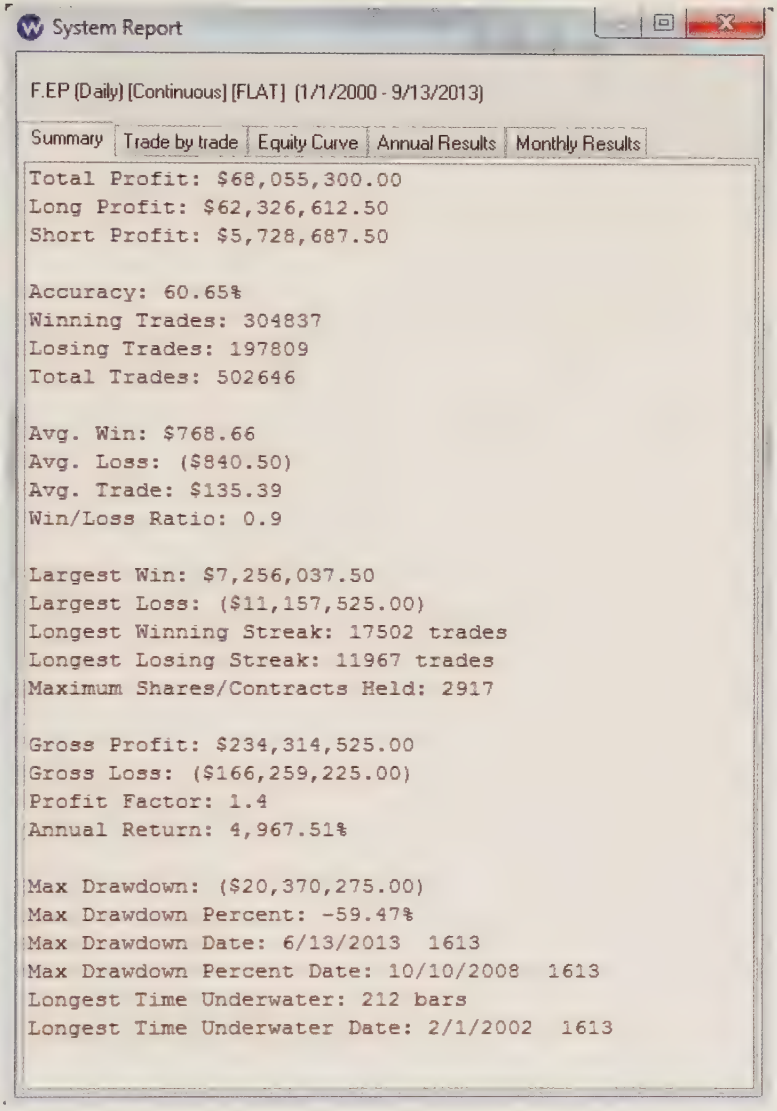


Table 11.5: Voting System with Aggressive Position Sizing System Report

$100k in to $68 million?? Yes, that's not a misprint! Unless you are getting in at the absolute ground floor in some groundbreaking new industry, you won't be able to find opportunities to earn that sort of return on your investment anywhere else. And all of this comes from the voting system we developed over the last few chapters. There's a reason Einstein said that compounded interest was the Eighth Wonder of the World.

This statement is nowhere as obvious as is it when working with position sizing on mechanical systems.

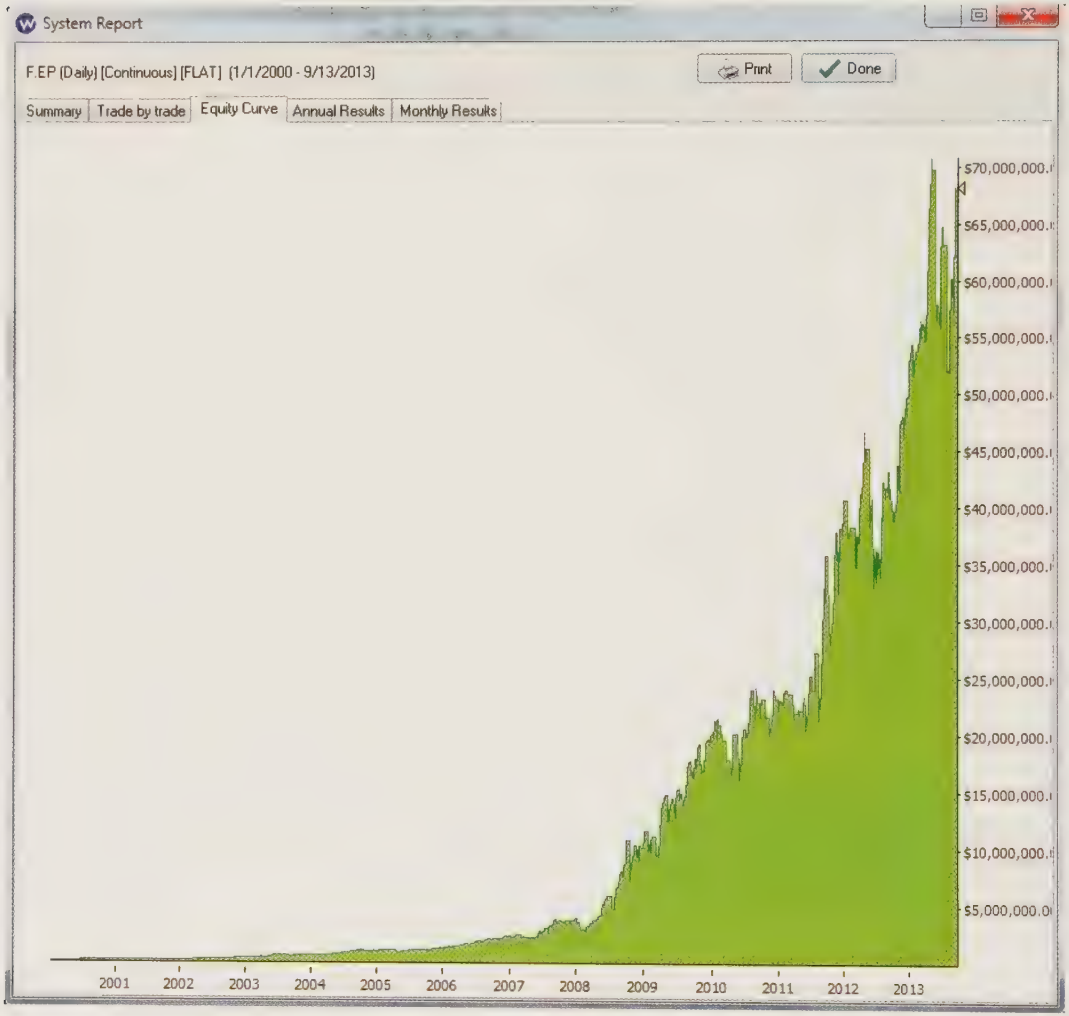


Table 11.6: Voting System with Aggressive Position Sizing Equity Curve

How can I not include an equity curve plot for something like this? Definitely a wild ride, and there was a $20M drawdown in there. But if you are looking at this opportunity in the right light, that should excite rather than scare you.

As we're on the topic of aggressive position sizing, there is another way to do this. Rather than basing our one-contract dollar value on the maximum drawdown, we can

base it on the largest loss. The idea is that each trade needs to be viewed independently, and all we need to worry about is to be able to survive until the immediate next trade. To do that, we need to have enough to cover our single worst losing trade, plus margin.

In the case of our voting system, our largest loss ever was a hit for -$7,975. Like drawdowns, we need to assume that there's a worse loss just around the corner, so let's add 50% to that number to arrive at $11,962, plus our $5000 margin requirement to get to $16,962 needed to trade one contract.

Keep in mind that this particular approach is only going to work if we can start trading more than one contract. That's because the account itself is still going to need to absorb the maximum drawdown without going belly up, and that isn't going to happen if we only deposit $16,962 in the account thinking that it is enough to trade one contract. Remember that our maximum drawdown number is -$15,612, and our margin requirement is $5000. If you don't have enough margin available in the account after a drawdown, you can't trade.

Assuming the worst case scenario, that we hit our max drawdown immediately upon starting to trade this system, and that the first loss in our drawdown is also our largest loser ever, the system is going to need to be able to scale down in size enough to weather the storm, and still be alive to trade after we've bounced back. So if you have a small account, but still want to be aggressive, use one of the previous methods instead of this one. But if you can start out with a larger initial account and trade multiple contracts right out of the gate, you can really ramp things up with this approach if you are ultra-aggressive.

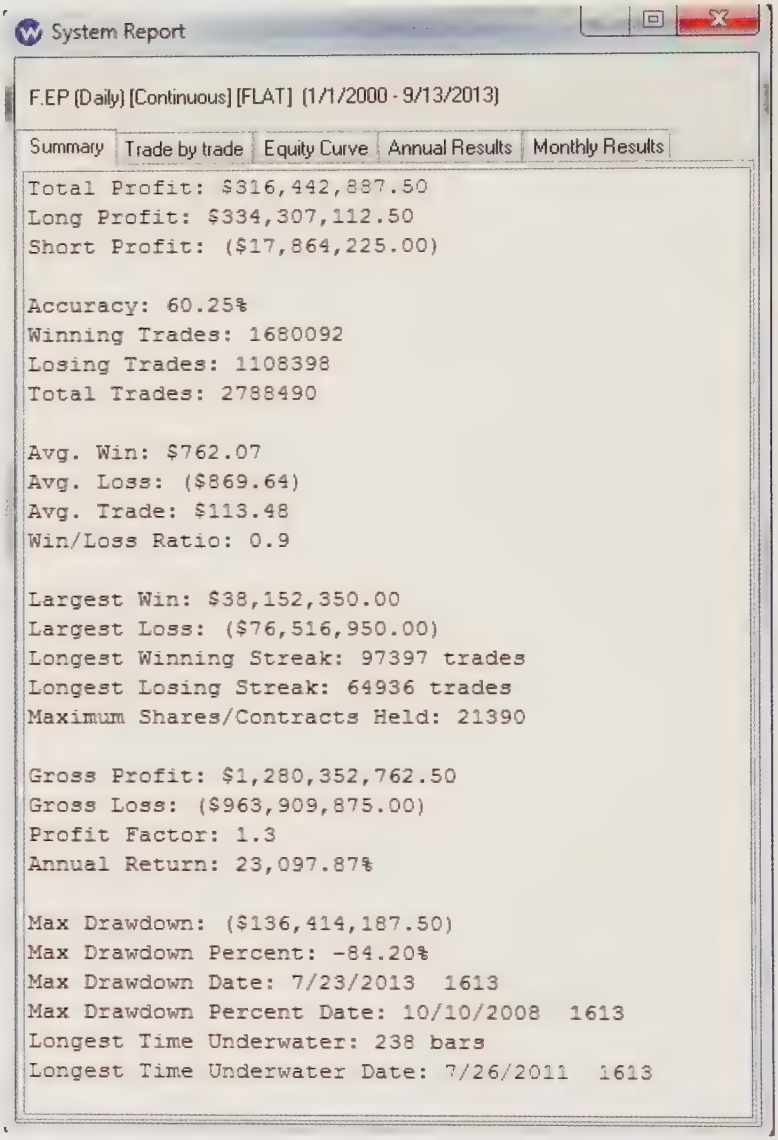


Table 11.7: Voting System with Ultra Aggressive Position Sizing System Report

Here are the results with the same starting account size of $100k. The numbers are starting to get insane. In this case, at its peak, the system was trading over 20,000 Emini contracts, which is enough to start moving the market around. So the results are probably not really doable in a practical sense, but they do tell you that as long as you're willing to be crazy aggressive, and if you've got a stable system to work with, it is

possible to grow an account as large as you can possibly want. If you are big enough to start moving the market with your order placement, let's just call that as successfully having navigated the trading game and leave it at that. I don't know too many traders who could pull something like this off, but an example like this is important because it shows what can be accomplished given the right goals and temperament.

I haven't covered all the position sizing systems available to traders, but I have given you the ones that I've used myself in my own trading. This is definitely an area worth studying more, but unfortunately a lot of your work will have to be done on your own here, as there are really only a handful of books available on this subject. As with all things trading, the most important topics are also the least talked about.

At the very least, you are going to want to spend some time working on how your system deals with the size question. Usually, this is done after the system has already been built, sort of as a way to polish it up. That's because you need to have a complete system available before you can do any work on the size part of the equation. Position sizing can add a lot of volatility to a system, and that's a good thing if the volatility is working on the profit side. But it will accentuate drawdowns as well, so before you start going crazy with position sizing algorithms, you need to make sure you have all your ducks in a row with regard to having built your system out to be as robust and profitable as you are able. Only then do you want to start messing around with scaling things up.

Personally, I would never trade a system without first having worked out how I was going to manage my order size. It's not something you can ignore, as the broker needs to know how many shares/contracts you want to trade, and you need to make sure that this doesn't turn into the Achilles' heel of your trading method. A fantastic system with no position sizing is going to look like a loser when compared to a good system that includes position sizing, and if you're going to be trading at all, there's no reason not to maximize your edge in as many ways as you can.

12. Options

In the last chapter, we explored a few methods that allowed us to transform a relatively small account into a gigantic one. This won't happen overnight, but if you give a trading system ten years to run, some pretty amazing things can happen if you apply position sizing appropriately. In the case of our most aggressive method of compounding that we used at the end, we're talking about huge, huge returns. But in order to implement all of those approaches, you're going to need to be able to set up an initial account large enough to weather inevitable drawdowns and difficult periods. If you start with too small of an account, you can almost guarantee failure, as the account will be blown after the first drawdown period.

Trading with too small of an account is probably the main reason why so many beginning traders fail when they get into the markets. You can have the most fantastic system, but if you can't keep trading through the first losing period, none of those profits are going to end up in your account. Not everyone has the $20-$25k required in order to begin trading futures, let alone that amount to throw at a super aggressive position sizing approach that can be expected to lose over 50% of its value at some point due to a bad drawdown. The worst thing to do in this situation would be to try trading a system with less than the appropriate amount of capital. That's a sure recipe for disaster. A better alternative is to steer clear of futures, and trade something that requires less capital to fund your account.

Stocks are a great choice here. Systems that work on the Emini futures market will also work on the SPY, which is an ETF designed to mimic the S&P 500 market, and can be done so at a fraction of the risk. Here's our merged system trading just a single share of the SPY:

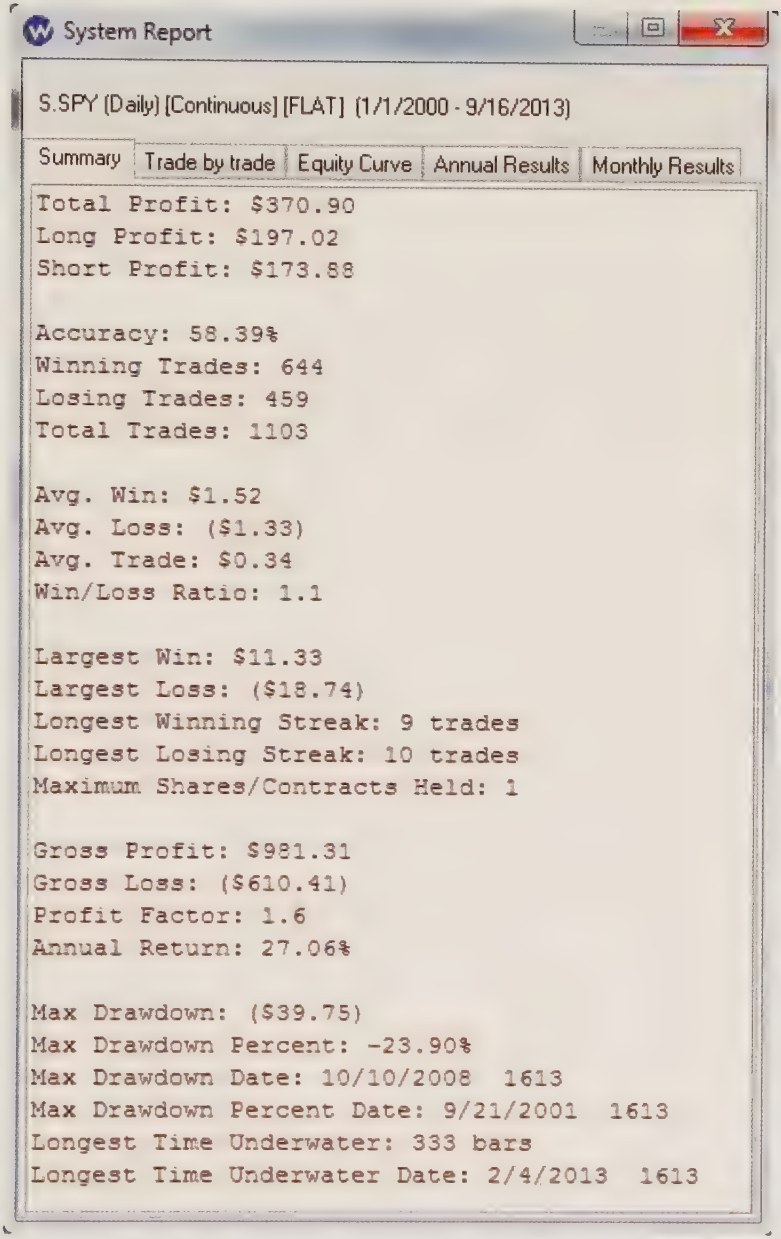


Figure 12.1: Voting System Trading a Single Share of SPY

This system has been run over the same period as we looked at on the ES. Right now, the SPY is trading at around $170/share, so trading ten of them could be done with less than $2000 in the account. You're not going to be able to retire on those results, but the

account will grow, and eventually will be large enough to transition into the futures market.

The main reason why I've never traded stocks is that they don't offer nearly the leverage found in the futures markets. Although you can use margin to gain some leverage, there is a legal limit to how much your broker can lend you in order to buy or sell securities. For the purposes of comparison, let's assume that we use as much margin as we are legally allowed, and trade the SPY with a $100k starting account size back in 2000, the same way as we approached the ES in the previous chapter. On every trade, we will buy or sell as many shares as we possibly can, given whatever our account value is at the time. So basically, this is as crazy as we can possibly trade according to legal limits.

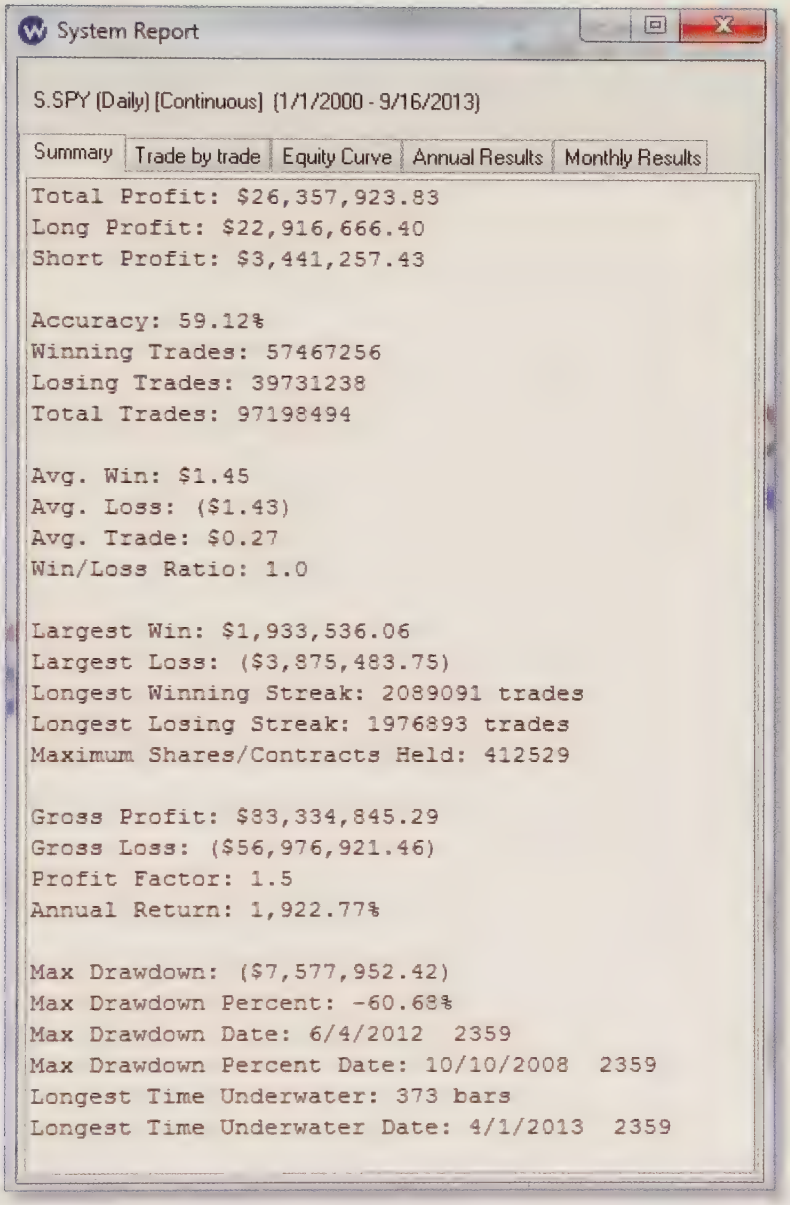


Figure 12.2: Voting System Trading the Maximum Theoretical Size for SPY ($100k)

You can see that the account earned $26M at the end of this period, which on its own seems like a great result. However, remember that with futures, trading as aggressively as we possibly could, this same approach earned over $316M in the same period, which

shows that you gain around 10x the leverage in the futures markets as you can ever hope for in stocks. Additionally, you're not going to have to worry about paying dividends when you're short, won't have to pay interest to your broker for your margin, and won't have to deal with margin calls when your positions go against you. Remember that we're maxing out our leverage in this simulation, which means there is no money left over to cover margin requirements should things go against us. So consider this to be the maximum theoretical return that trading the SPY could generate.

Traders with large accounts have lots of ways to earn a good return. Practically speaking, had you invested $100k in either the SPY or the ES aggressively as in either of these two examples, you'd be pretty happy with the end result. Sure, the ES trader made almost $300M more than the SPY trader, but the SPY trader still has over $26M to show for his efforts, and that's nothing to sneeze at.

Keep in mind that both of those traders had $100k to invest at the commencement of trading. The problems arise when we start running these simulations out using much smaller account sizes. If we are going to be trading the SPY rather than the ES with this system, the reason for doing so is likely to be that we just don't have enough capital to trade even one contract on the ES. So let's keep things realistic here, and start the account off with only $5k back in 2000 and do this again.

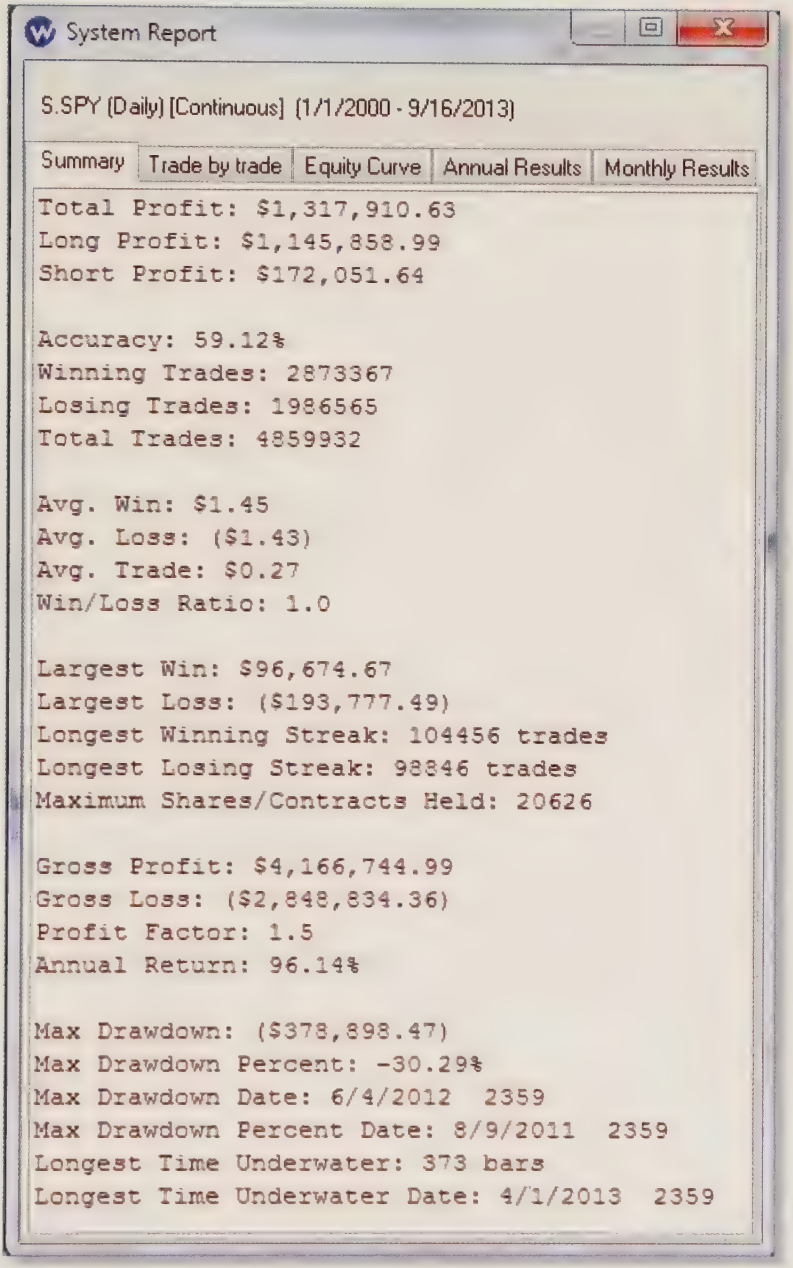


Figure 12.3: Voting System Trading the Maximum Theoretical Size for SPY ($5k)

In this case, our results are still good, but we're not going to be buying any jets soon. Even with the great percentage return at the end, the big problem is that the money

doesn't grow fast enough. We had to wait all the way until 2009 before the account broke the $100k threshold, which is a long time to be plugging away at a trading system.

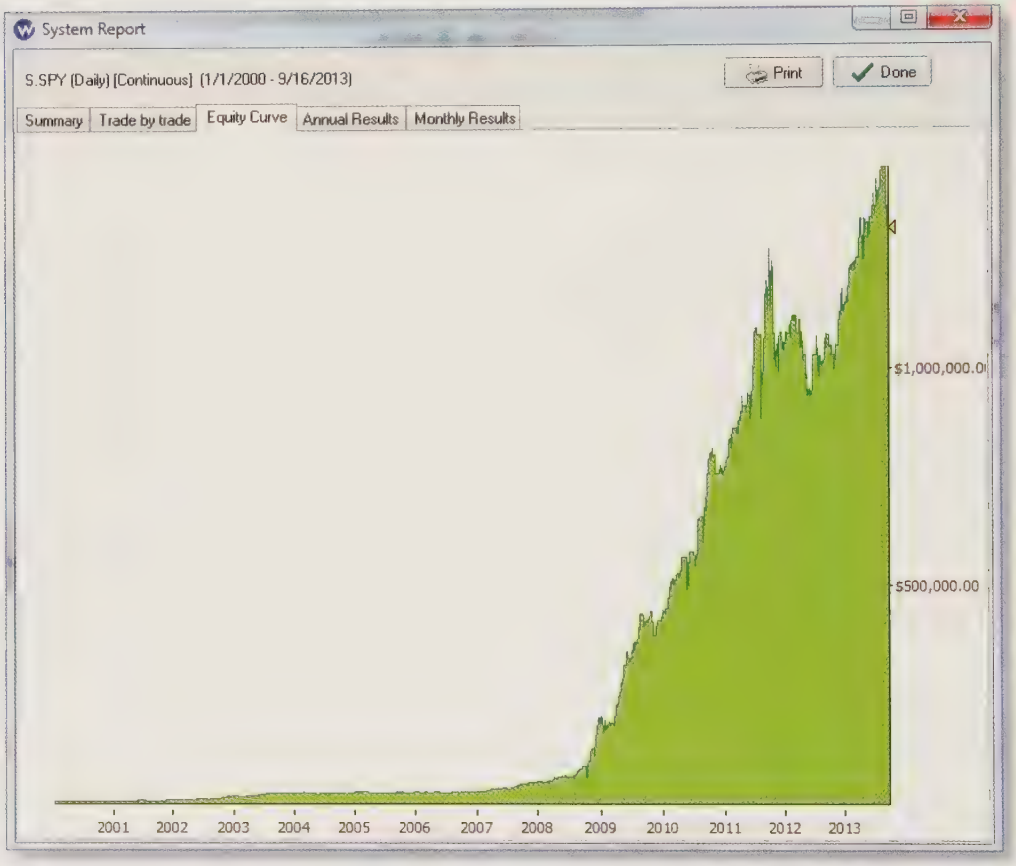


Figure 12.4: Voting System Max Theoretical Size for SPY ($5k) Equity Curve

If you are just a hobby trader, this might be fine, but if you are trying to break into the professional category, or are interested in leveraging your account up into the multi-million dollar realm, this is too slow. $100k is where we started the previous run, which means that in order to duplicate it, you need to add nine years to the length of time required. Personally, I don't want to have to wait 20 or 30 years to get to that stage. Remember, systems all have a life span, and we don't want to have to count on this one running for 20 years in order to get our big payoff. We need this to happen faster, not only for that reason, but also because we want to be able to use this money to enjoy life while we are still young enough to do something with our profits. If you're already dead

and gone by the time your system attains the levels you were waiting for, that actually counts as a failed system, among other things.

So if you don't have enough capital to trade futures, and you don't have enough patience to trade stocks, is there another alternative? There is, in the form of the options market.

There are tons of options books available, and a hundred different options strategies that you'll hear authors talk about: spreads, straddles, strangles, iron condors, butterflies, etc. There are endless numbers of combinations of options positions that all result in interesting profit profiles that can be taken advantage of by traders who know how to use these vehicles. There is a lot to know about options trading in order to do it properly, but for our purposes, we can ignore most of that. What we are really interested in is using options as a way to gain leverage when trading our system.

At the most basic level, there are two kinds of options: calls and puts. Call options allow you to purchase a stock at a fixed price, and put options allow you to sell a stock at a fixed price. If you think the market is going to go up, you would buy calls, and if you think the market was going to go down, you would buy puts. Each option controls 100-shares of stock, which means that theoretically in the case of the SPY, you can control around $17,000 worth of stock for only a few hundred dollars. If we trade in options instead, we will be able to control quite a bit more stock than we had been able to in the simulations that we just ran, which should help with our problem of how to begin trading this system using a small account.

Of course, the last thing we are going to do is run out and start purchasing options using our SPY signals. Like everything else we've done, we have to prove that this is going to work before we invest any real money in the approach. That means we need to take our system and simulate the results as if we had traded options rather than stocks, which is easier said than done. Historical options data isn't as easy to come by as stock or futures data, and for every optionable stock, there are hundreds of options trading at exactly the same time, each of which expires at a certain point in the future. So not only is there a huge amount of data involved, it is also not clear which options specifically we need to be trading.

As it turns out, earlier this year I spent some time building out a SPY system for myself. We've already seen details from its system report in Chapter 2 when I used it as the example system when going over all of the metrics. After I had completed development on that system, I ran similar simulations to what we just looked at, and realized that I had an issue in the leverage department. I wasn't going to trade the SPY itself since I just wasn't getting the bang-for-the-buck I was used to having been involved in futures for so long, so I needed to figure out how to implement this method using options instead. What I'm going to share in the rest of this chapter is the work I did getting that method ready for options trading, as well as some of the surprising things I learned when working with the options data.

The first issue I came across when trying to implement my system in the options market was the data issue. In a perfect world, I would have access to all the options data available on the SPY, and would have the computer go through it according to my rules and spit out various system performance metrics in the same way it does when running on stocks or futures markets. Unfortunately, life isn't that easy, but I managed to gain access to an options brokerage account that provided 5 years of SPY options data in tabular form. So I spent a few weeks building up spreadsheets of options trades by hand to see how well my system performed when trading options directly instead of the SPY.

Doing system testing by hand is never easy, and this is not an exercise I'd be interested in repeating any time soon. So I hope you'll forgive me for switching up systems on you at the very end. The conclusions I derived from my own tests should be equally applicable to using options in conjunction with our merged system, and I really want to pass this information along to help others steer clear of some not-so-obvious pitfalls that arise when attempting to implement a stock trading system using options.

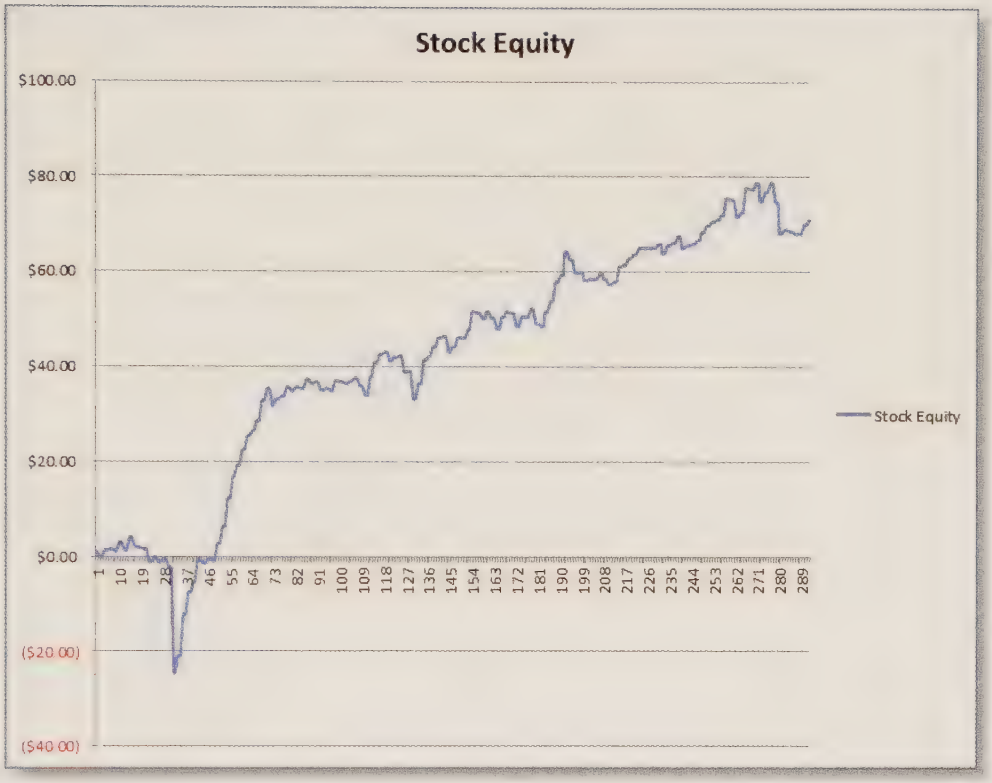


Figure 12.5: Equity Curve Trading SPY

The first chart we will look at (Figure 12.5) is the equity curve from trading the stock system from 4/25/2011 through 8/6/2013. There's nothing special about those dates, that's just the period I ended up using for this test. So a little over two years. I'm not trying to prove out the system here, I have already done that. I'm trying to see how I can substitute options for shares of stock. Ideally, I want to be able to get the exact same equity curve, but using a smaller required account size. The graph above shows the result of trading one share of stock over this period. The system had a bit of a drop in 2011, but has been doing well overall. We'll use this as our baseline.

Next, I used the signals generated from the stock system, but rather than buying or selling shares of SPY, I placed options trades instead. I tested the following possibilities:

1) Buying 2-strikes in-the-money
2) Buying 2-strikes out-of-the-money

3) Buying 4-strikes out-of-the-money
4) Selling an at-the-money option

For example, if we were trading according to the first possibility, a trade would look like this. Let's assume we have a buy signal, and the SPY was trading at 167.23 at the time. We would first round the price to the nearest dollar, in this case, that gives us 167. Since this is a buy signal, we are going to be purchasing call options. A 2-strike in the money call has a strike price of 165. I would trade options with around 2-3 months until expiration to avoid losing too much of the purchase price to time decay. Rolling over would happen arbitrarily on my end whenever I felt it was time to do so. So if it were currently late April, I would likely be looking to purchase the July 165 call. I would enter the purchase price in my spreadsheet and then move forward until the date of the exit signal. I would look the option price up again for that date and would record a profit or loss accordingly.

Possibilities two and three work exactly the same way, except for the change in the strike price. In the fourth possibility, rather than purchasing an option, we are going to sell one. So if we had a buy signal from the system with the SPY at 167.23, we would sell a put option with the 167 strike at that time. The idea here was to test options writing in comparison to options purchases. To keep things orderly, I used the same expiration month for all options.

Here is a plot of the equity curves for all four of these possible scenarios:

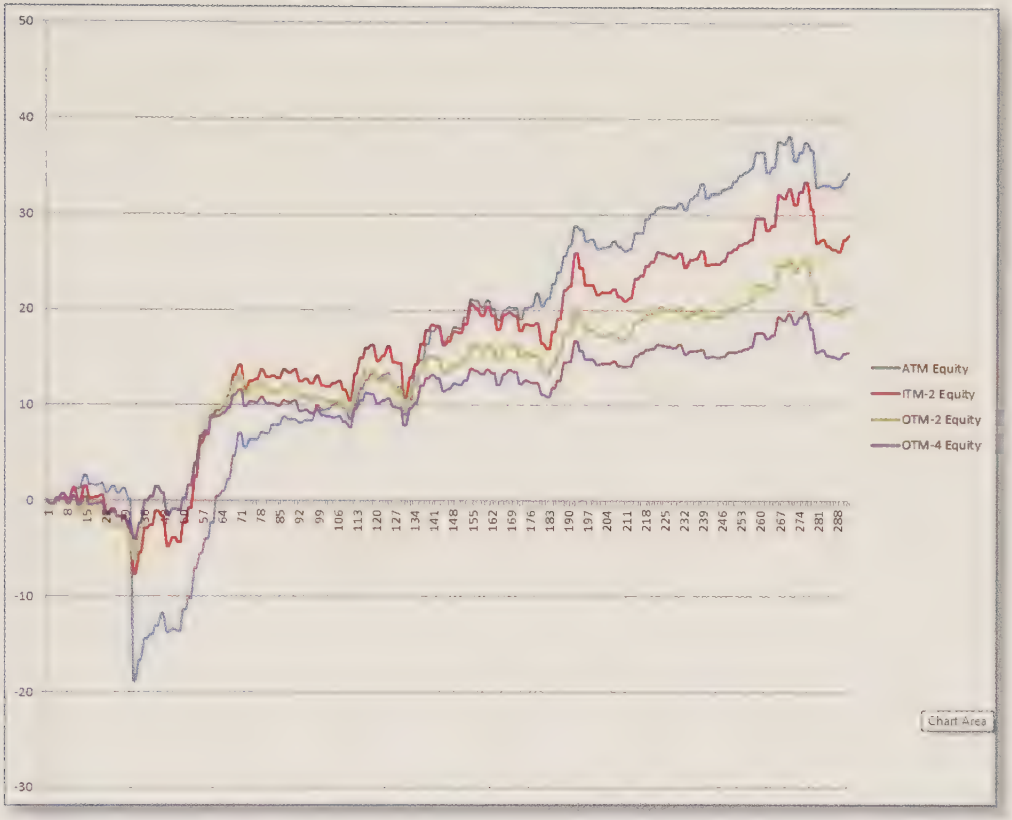


Figure 12.6: Equity Curve for Option Strategies

Although options trade in 100-lot size, these results show the gain/loss at a 1-share basis to keep everything level. There are a number things that become clear very quickly after studying these results. First, the most money was actually made by writing options rather than purchasing them. This is shown by the blue line in the chart above. Not only does writing options result in a higher profit at the end of the period, the equity curve is generally smoother. You can see this in the graph, especially around trades 70-110, where the blue line increases while the other three lines decrease.

The reason for this discrepancy is that when we write options, time decay is on our side. This is most obvious when we go back and look at times when the trading system entered a position and then exited it at almost the exact same price. In a situation like that, all the options purchases will have lost money due to time decay, but the options

selling will have gained money for the same reason. So on average, writing options results in more profits per option, as well as a better looking equity curve.

The problem with writing options can be seen towards the left of the graph, where the system went into drawdown. You can see that the blue line dropped significantly lower than the other lines there. That's because option writing has limited upside, but unlimited downside, and that's not a good thing during market crashes. This disadvantage becomes very clear when we plot the ATM option writes alongside the outright stock trading equity curve:

Figure 12.7: ATM Options Sold vs. Stock Equity Curve

You can see that our one option trade accounted for almost the same amount of drawdown as the outright stock trade. So we have equal downside, but clearly not equal upside, as the stock trading far outperforms the options selling in the long run, by a factor of almost 2-to-1. In other words, using options writing as a replacement for stock trading results in a situation where we can earn the same amount for twice the risk. That's the opposite of what we really want to be doing.

I did try transforming the naked option writes into a spread, to cap the downside potential. As it turns out, the money lost on the protection is larger than the money lost due to taking those hits, which means we'd earn even less using that approach than just sucking it up and writing those naked options. The approaches that seem to make sense at a theoretical level can quickly fall apart when backtested using real data. So at this point, we can cross the ATM options writing off our list of possibilities. That one isn't going to help us.

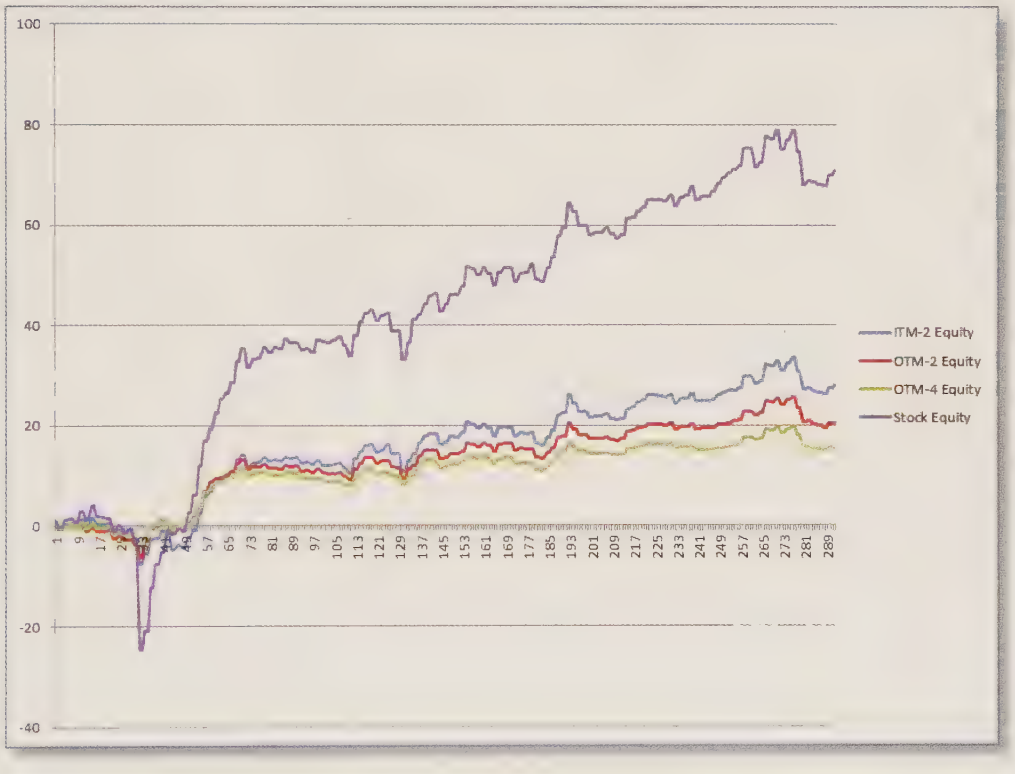


Figure 12.8: Option Buying vs. Stock Equity Curve

Now we're left with the three option purchasing approaches, as well as our original stock system. They're all shown in Figure 12.8 above. Although an options contract technically controls 100 shares of underlying stock, that isn't really the case when trading in the short term like this, and you can see how much more money outright stock purchases make than options trades. So let's figure out exactly how many shares an options contract really lets us control, from the perspective of trading this system.

We know the ending value for all of our equity curves, so we can simply compute how many options contracts we'd have had to be trading in order to replicate trading 100-shares.

Options approach	Number of shares each option is worth	Contracts needed to replicate 100 shares
2-strike ITM	39.37	2.54
2-strike OTM	28.82	3.47
4-strike OTM	21.98	4.55

Table 12.1: Stock Equivalency for Option Strategies

Although the options educators tell us that one option contract is the equivalent of controlling 100 shares of stock, that's not going to be true unless the options are very far in the money. These numbers are similar to an option's Delta, meaning how much an option will move if the underlying stock moves exactly one dollar, but aren't exactly the same. Our numbers have been calculated as the result of a system backtest, and would have been different had our system acted in a different way. A long term stock system will likely give different results here than a short term one, so you really need to compute this on a system-by-system basis. Having said that, the system I ran this on trades just about as frequently as our merged system, so I'd be surprised if these numbers weren't very close to the numbers we'd get when backtesting the merged approach.

Now we have three ways to duplicate the results of trading 100-shares. Which way is best? In addition to trading results, we can also compute the average price spent on each option contract. In our case:

Options approach	Average option price	Average price to replicate 100 shares
2-strike ITM	5.37	13.64
2-strike OTM	3.25	11.28
4-strike OTM	2.44	11.10

Table 12.2: Average Option Price for Option Strategies

The purchase price of an option determines not only how many options we can buy, but also represents our maximum risk on a trade. Therefore, all other things being equal, it's better to go with whatever method results in the smallest cash outlay. In this case, the

4-strike out-of-the-money option is the clear winner. We need more of them to replicate trading 100 shares, but they make up for that by being cheaper.

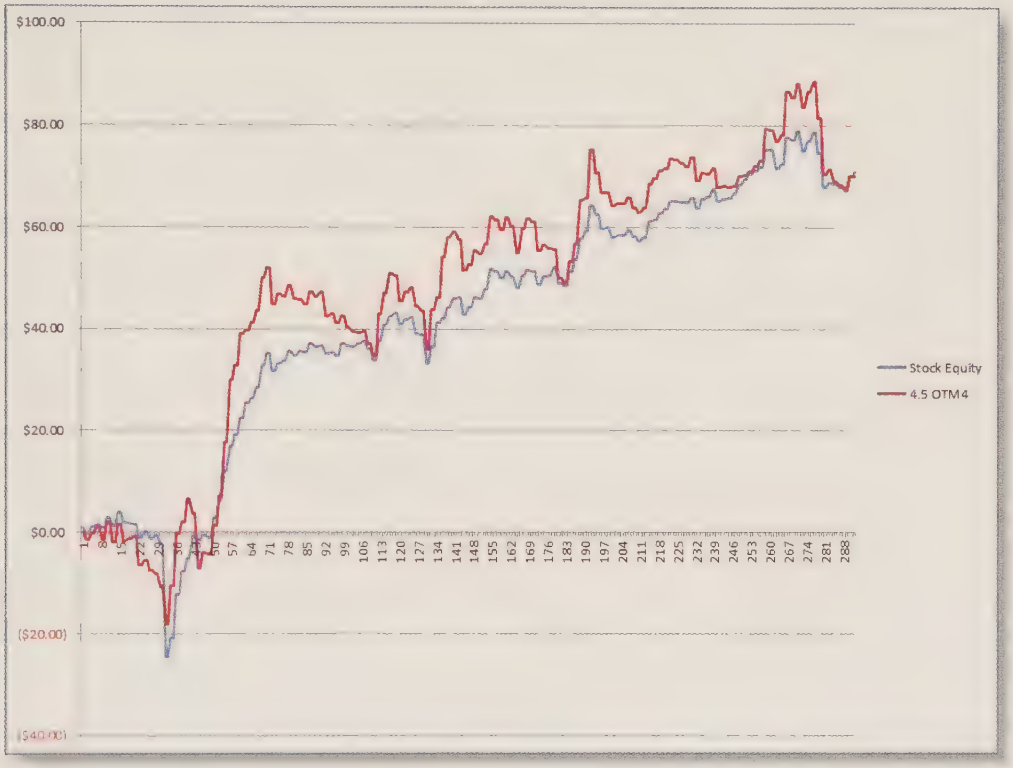


Figure 12.9: Equivalency of 4.5 * OTM4 vs. Stock Equity Curve

This chart shows two relative equity curves. The blue line is our underlying system, trading the stock itself, and the red line is our OTM4 system, trading 4.5 options contracts. The two equity curves are roughly similar, although the options curve is clearly a wilder ride. Time decay and volatility changes affect option pricing in ways that the stock itself is immune to, and you can see the result – the upswings are fast, but so are the downswings.

So now we know, in general, how to go about using options to trade our stock system. Let's take this a step further and add position sizing. We already know how to do this with the stock itself – we decide how much margin we want to use, and we simply buy or sell the maximum number of shares available given our account size at the time of the trade.

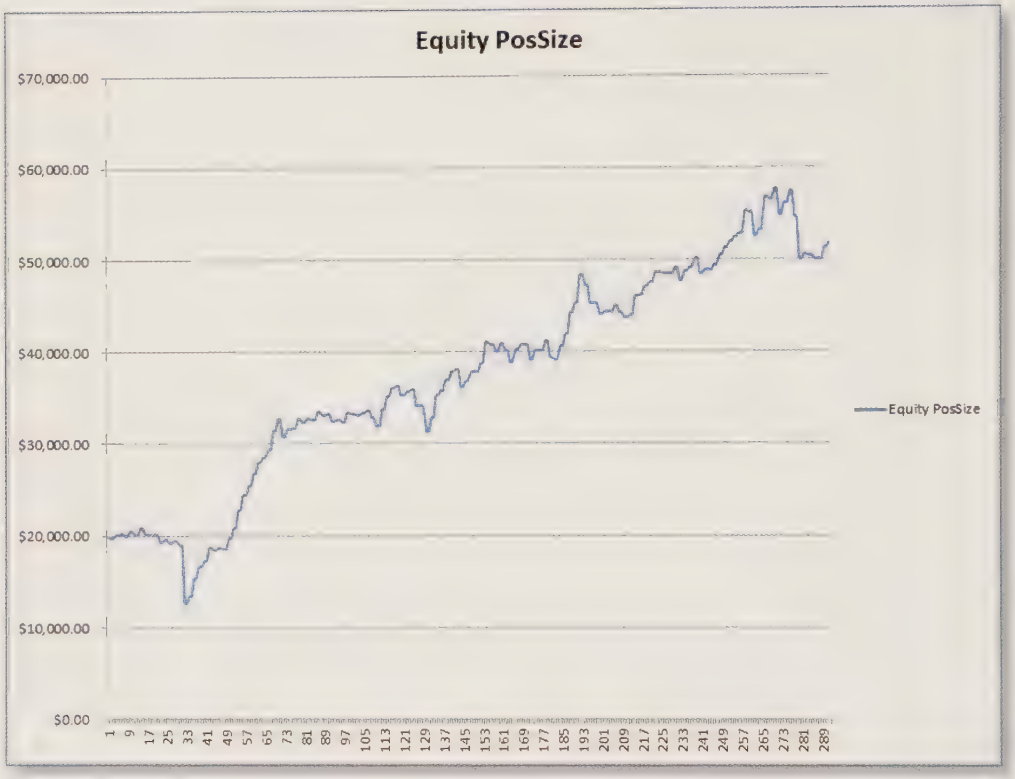


Figure 12.10: Stock Equity Curve Trading the Max # of Shares Using 50% Margin

With the most aggressive possibility of 50% margin, a $20,000 account would have had a run as shown in the graph above. A 150% return on account in just over two years is nothing to be ashamed of – this is a good system, and it can be leveraged up quickly using margin. What we want to do is beat this result using options.

The funny thing is that when I was running these tests, I was already trading this system using options. I was impatient, and wanted to get some live experience with the system. I knew that I wanted higher leverage than stocks provided, so I adopted the method that I'm sure is currently in use by countless other options traders working with these same issues. Specifically, I designated a specific percentage of my account that I wanted to risk each trade, and would purchase as many option contracts as I could without going over that number.

Let's assume our account value is at $20k, and we want to risk 5% of it. That means on the current trade, we are allowed to risk $20,000 x 0.05 = $1,000. If our options are trading at $200 each, we would then be allowed to purchase five contracts. If we are trading the 4-strike OTM options, that is going to be almost like trading 100 shares, but we can do so with risking only $1000 rather than the entire account.

Figure 12.11: OTM4 @ 5% Risk vs. Stock Equity Curve

You can imagine my surprise when I plotted the chart shown above. The blue line is our same 50% margin stock trading system, and the red line is the result of risking 5% of the account on options purchases. That strategy is going nowhere! Let's up the ante. I was planning on getting more aggressive as soon as the backtesting was done, so let's bump that number up to see what happens.

Here's the comparison when risking 20% of the account:

Figure 12.12: OTM4 @ 20% Risk vs. Stock Equity Curve

In this case, we suffered a loss equal to our outright stock position early in the testing period, but are still way underperforming when compared directly to the stock trading system. Not exactly what we were expecting here. Let's bump our account risk up to 50%, which is super aggressive and see what that does.

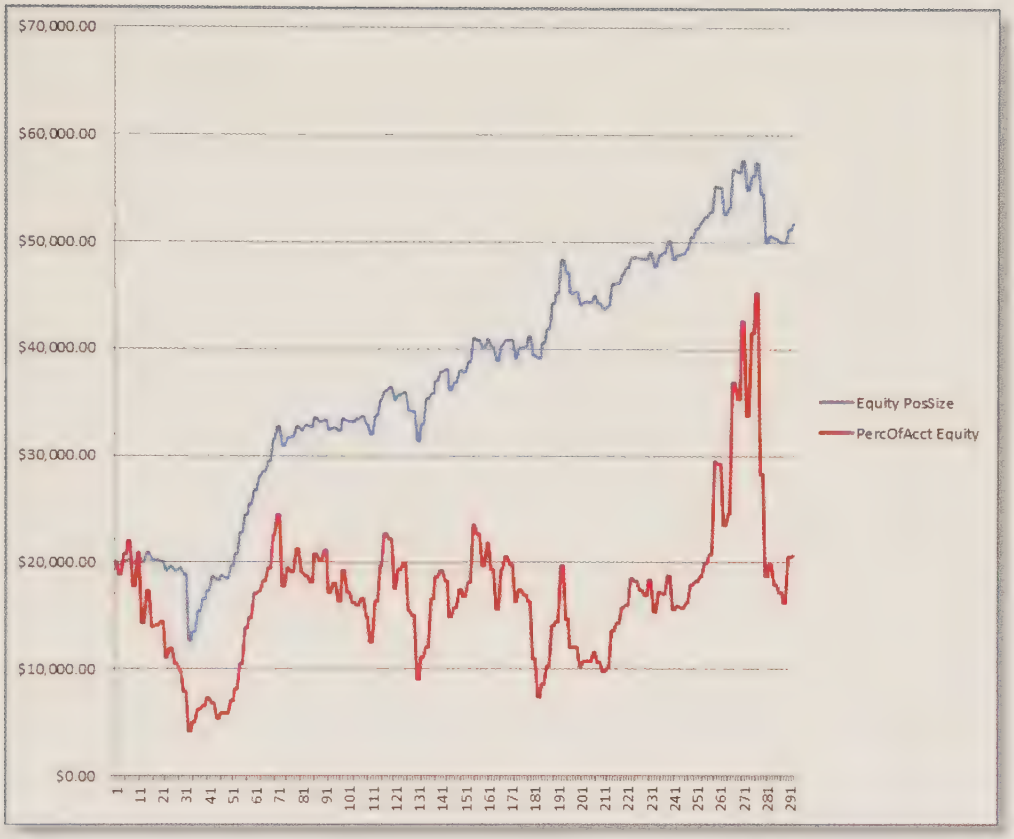


Figure 12.13: OTM4 @ 50% Risk vs. Stock Equity Curve

You can see that our results are actually degrading as we increase the risk. This should be the equivalent of trading a huge amount of stock, and we would expect to see an explosive equity curve taking us to astounding levels of profitability very quickly, but we're not. In fact, risking 50% of our account resulted in us barely breaking even after the two year run, and this is using a system that is almost 70% accurate and has been crushing the underlying.

So what is going on here? Why are we unable to match our stock returns when using options? The problem lies with our percentage of account risk model. When option prices are very cheap, we end up purchasing lots of contracts with this approach. Conversely, when options prices are very expensive, we purchase a much smaller number of contracts. Not all options are equal, and as it turns out the probability that an

option will become profitable is built into that option's price. More expensive options are more likely to increase in value in the short term, and less expensive options are less likely to increase in value. Like a lot of other things in life, in the options market you get what you pay for.

So what we really want to do is buy more contracts when the options are expensive, and fewer when they are cheap. Exactly the opposite of what every new option trader is doing! Makes sense from that perspective though, doesn't it?

Here is the formula I came up with to determine position sizing for options-based systems. It's somewhat involved, so we will step through it. First, we must calculate the base size:

$$\textbf{Base Size} = Contracts\ per\ 100\ shares * \frac{Account\ Size}{\left(\frac{System\ Max\ Drawdown}{Desired\ Drawdown\ Percentage}\right)}$$

This amount tells us the theoretical number of options we need to trade, to match what our stock system is doing. System max drawdown divided by our desired drawdown percentage gives us the account size we need to be working with in order to trade this system. This comes straight out of Chapter 2. In the case of my system, my drawdown was -$3037 trading 100 shares, and I don't want to drawdown more than a third (33.3%) of the account. That gives me a required account size of $9120. If we divide our account size ($20,000) by this number, we arrive at 2.19, which is the number of 100-lots we need to trade to keep drawdown appropriate. Since we're trading 4-strike OTM options, we need to then convert that number to options contracts rather than shares. We've already calculated that we need to trade 4.5 OTM4 options to replicate a 100-lot, so we then multiply 2.19 by 4.5 to arrive at a Base Size value of 9.85.

So what we are doing is going back to our stock trading system, forgetting about margin or any of those things, and figuring out how many shares we should trade to keep our drawdown to a respectable limit. We then figure out how many option contracts we need to trade for the equivalent risk.

Next, we are going to adjust our size based on the option price. Since we know that cheap options are more likely to cause losses in our account, we want to purchase fewer options when they are cheap, and we want to purchase more options when they are

expensive. How do we determine cheap or expensive? We figure out the average price of an option, given our system and approach. In my spreadsheet, 4-strike OTM options trade at an average price of $2.44. Therefore, if an option is trading at a higher price than that, it is expensive, and if it is trading at a lower price, it is cheap.

This brings us to the next calculation:

$$\boldsymbol{Num\ Options\ to\ Trade} = Base\ Size * \frac{Options\ Price}{Avg.\ Options\ Price} * Multiplier$$

We take our base size value as just computed, and we adjust for an option being expensive or not. This is simply the ratio of the option price to the average option price. Larger numbers will increase our position, and smaller ones will decrease it. I then added a multiplier, which is a way to make this more or less aggressive.

To complete our example, if our base size value is 9.85, as we have already computed on the previous page, then in order to figure the number of options to trade today, we'd need to know today's options price, as well as our average price. Let's assume the option we want to purchase is trading at $2.11. We already know our average OTM4 option costs $2.44 from the previous table, so assuming a multiplier of 0.5, which is what I usually like to use, then:

$$Num\ Options = 9.85 * \frac{2.11}{2.44} * 0.5 = 4.26 = 4$$

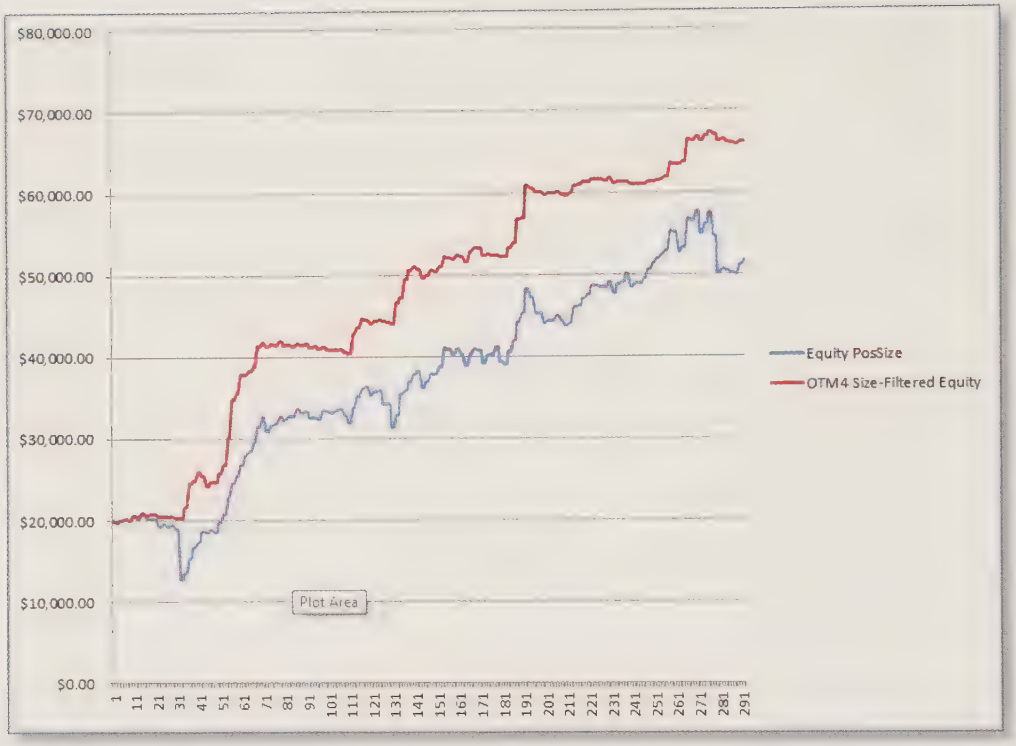


Figure 12.14: OTM4 Price Adj. Position Sizing (multiplier 0.5) vs. Stock Equity Curve

This is the result of our new formula. Not only are we making significantly more money, we are also doing so at less risk! You can see that the red line, which is our options equity curve, tends to go flat from time to time. This correlates to periods when the stock system also goes relatively flat, which would happen during very low volatility times when options prices would likely be relatively cheap. Although the system is always buying some options, it buys fewer then, and avoids a lot of time decay. Then, during volatile periods, it increases its size, as options prices would be higher, and the result is that the big streaks upward get exaggerated.

This graph uses a multiplier in the formula of 0.5, which resulted in us never taking on a position larger than 19 options contracts during the entire test. If we want to be more aggressive, we can do so with a higher multiplier. Here's what 0.75 looks like:

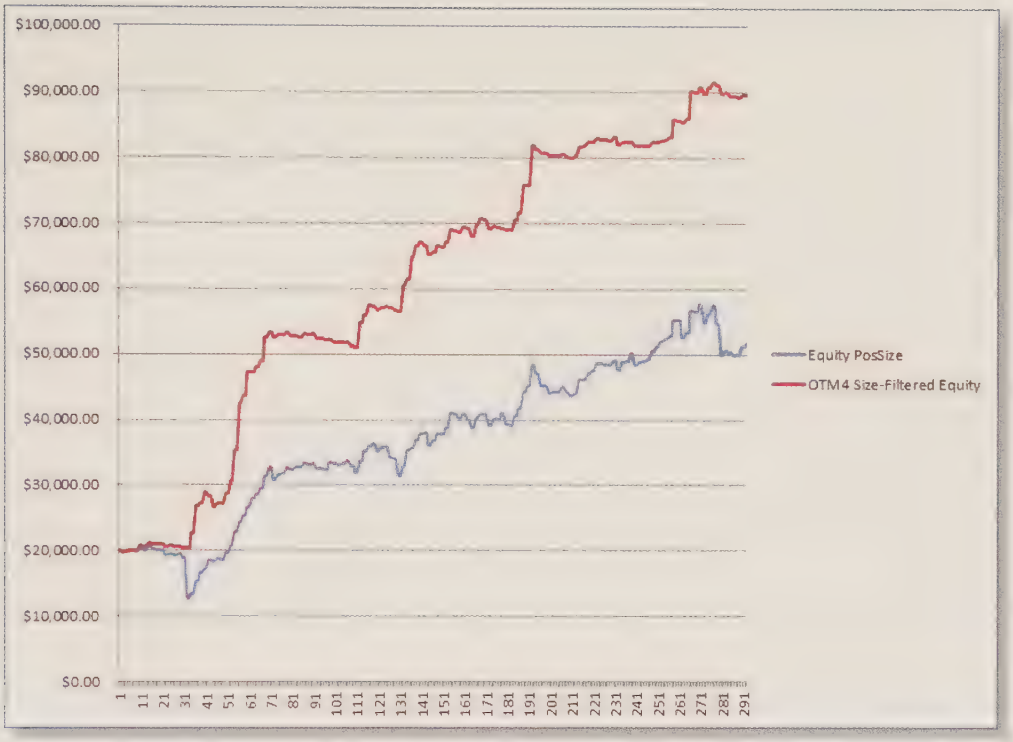


Figure 12.15: OTM4 Price Adj. Position Sizing (multiplier 0.75) vs. Stock Equity Curve

Now we're talking! Not only have we almost doubled the gain from the heavily margined stock method, we have been able to do so without increasing our risk. This then gives us the leverage we need to compete with the futures traders.

Although I only used a couple years of options data in order to test this, we can use our results to make an educated guess about how our newfound increased leverage can be used in the long run. My spreadsheet allows me to simulate different levels of margin trading on the original account, and a margin level of 30% is in the ballpark when it comes to trying to get the blue line to match the red one. The options curve still looks a little better because it is cleaner in general, but the two are roughly equivalent.

If we go back to our merged system, and run the account out on the SPY using a 30% margin number, meaning that we are borrowing 70% of our purchase and only contributing 30% of the price out of our own funds, we obtain the following result:

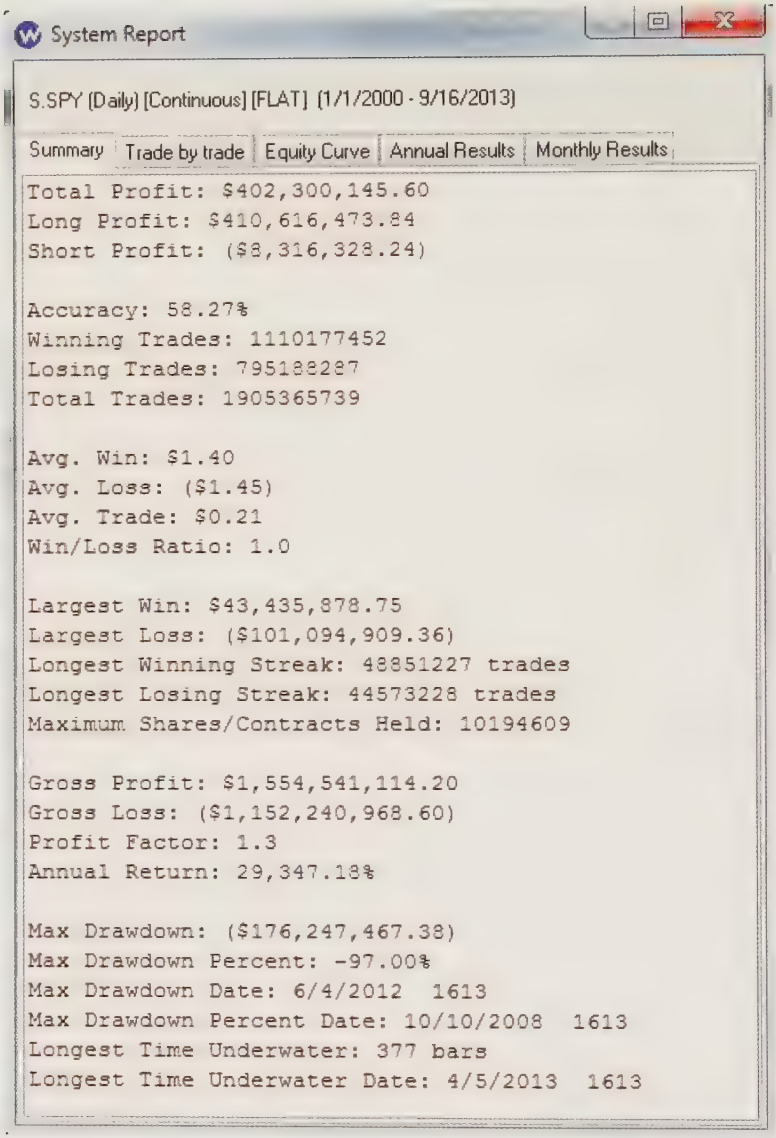


Figure 12.16: Proxy Results for the OTM4 with Price Adj. Position Sizing

As you can see, we now have a way to trade our system on stocks that gives us even more leverage than the futures market. Of course, we can't actually trade using this level of margin, since that's illegal, but it is meant to simulate what things would look like if we run our options method out over the same period of time we've been using to

judge all the other approaches. Like those other approaches, this report is based on a starting value of $100k in the year 2000, run up until September of 2013.

Practically speaking, the options markets aren't going to be able to provide the liquidity of the stock or futures markets that we'd need in order to grow this large, but we could trade a small account in the options market and then move it over to futures when it got too large. This gives us a way to aggressively trade a small account, which means you don't have to already be rich in order to have a chance at swinging for the fences using systems and position sizing. Congratulations, private islands are back on the menu for the stock traders..!

This chapter gives you an idea of how to use options as a replacement for stock when trading a mechanical system. Like many other things in the trading world, the most obvious way to do this (trading a fixed percentage of the account), is also one of the least profitable methods. The actual price of the option contains the probability that the option will make money in the short run, with more expensive options being more likely to generate profits. That means that we want more options when they're expensive, not fewer, and trading a fixed percentage of the account is going to result in doing the opposite.

Of course, the only way we would know this is by running a system through a full options backtest, as this information doesn't appear in options books. It shows the benefit of sitting down and cranking through the numbers, even if it has to be done by hand. If you don't do that, then you really have no way of knowing what your edge is, which means your system is based on guesses, and guesses tend not to be very good at generating profits in the long run.

If you decide to implement this approach yourself, I would strongly recommend doing an independent backtest in the same way I did. You will need to know the average price of your option at the very least, and rather than taking my word for it, it would be better if you figured that number out yourself. It will likely change over time, and you want these numbers to be based on the system you will actually be using for signals, not on mine.

One thing to be aware of if you decide to set up a similar study is that puts and calls don't trade at the same prices. Puts are always more expensive. Markets go down faster

than they go up, and I suspect a lot of traders use puts as insurance, so they just tend to cost more than calls. I just lumped them all together, but since there is a difference, it means that in the approach I'm using, our system tends to buy more puts than calls. Whether or not that is a good thing is unknown at this point, and it might be better to compare calls against the average call price, and puts against the average put price, rather than everything against a single combined price as I am doing. My method works, and the numbers back me up, but it just might not be the optimal approach. I'll leave further tweaking of this formula to the readership, and I'm certain there is a lot more here to discover. Let me know what you find!

13. Conclusion

Phew! We've covered a lot in this manual. The most important skill I wanted to try and pass on is how to look at systems with the appropriate critical eye. Had I possessed that skill early on in my own trading career, I would have saved myself quite a bit of grief. Hopefully you can learn from my mistakes and skip over some of the lessons I've had to learn the hard way.

Most trading systems will tell you up front whether they are about to blow up in your face or not. The most curve-fit ones practically scream it at you. But you need to be able to speak their language before you can understand what they are saying. Hopefully, by walking through these system reports, studying equity curves, and taking apart a bunch of my old systems, you'll have learned the basics of that language, and can take things forward from here successfully on your own. You now have everything you need in order to build your own profitable trading systems, and should easily be able to navigate around the most common pitfalls that sink most would-be system developers.

The most important thing to remember is that your system doesn't need to be the holy grail. What you really need is a system that is robust, that is based on something that really does provide a timing edge, and that you can trust moving forward into unseen data. A system like that can be transformed into the holy grail through the magic of position sizing, but none of that will work on poorly designed, over-optimized, curve-fit models. Those systems look fantastic in hindsight, but we have to trade the future, not the past.

If you put my random exit method next to a turbo-charged, multi-core neuro-genetic-fuzzy-super-optimizer that some non-trading PhD types put together, I'm going to get crushed in the backtests. There's just no way to out-optimize some of these modern optimizers, and they'll earn millions trading one contract on past data. But you take those two systems and put them into tomorrow's market, and the tables will be completely reversed. The optimized method will catch fire, and burn itself to the ground along with your account, while the simple random exit approach will just continue climbing along, losing a little here, and then making a little more there, just like it always has.

I did want to take a minute and talk about slippage and commissions, as I know some of the more experienced system builders have likely been getting worked up about that subject. In all the system reports I presented, we saw the theoretical value of the system, as if it could trade with no slippage or commissions. Obviously, that isn't real life, and we need to make sure that our systems can overcome the costs of trading. Although this will change in the 3.x series of Wave59, as of version 2.17, it doesn't include slippage and commissions in its system tester. Obviously, I've had plenty of time to add that feature, but never got around to doing so. Why?

When I develop systems, one of the key metrics I pay attention to is the average trade. This tells you how much your system can be expected to generate, on average, each and every time it trades. When I look at systems, I only pay attention to systems that have high enough average trades that slippage and commissions become irrelevant. On the ES, a commission is around $5 per round-turn, and you can expect to lose one tick on the entry, and one tick on the exit if you use market orders. One tick is $12.50 per contract, so on average, you'll be paying $30/trade in costs. If your average trade is $250, will those costs really be an issue? No. But, shouldn't we include them anyway, just for completeness sake?

If I've tried to beat one thing into the heads of the readers of this book, it's the fact that all parameters migrate over time. The perfect moving average length might be 10 in one period, but in the immediate next period, it will shift to 25. This will happen whether you are trading indicator X or indicator Y, on any time frame, and on any market. It is the nature of markets to do this.

Systems are just a different kind of parameter, and aren't exempt from this phenomena. If you run a system on a specific data set, you'll get a certain result. If you run the system on a different time period, your result will be different. Your profit values will change, your accuracy will change, and your average trade values will change. This is especially true as you move into unseen data. So when you start trading with real money, those past results actually don't mean anything, as markets are completely new each day.

The purpose of system testing is to tell you whether or not your method is robust enough to make money when trading live. The actual specific numbers themselves don't really mean anything on their own, because they are guaranteed to change on you and

need to be viewed as estimates. If an average trade value is low, it means odds are you won't be able to make money. If an average trade value is high, it means odds are that you'll be able to absorb costs and make money. So I tend to think in terms of small/medium/large, rather than in dollars and cents with a lot of these things. That's also why I really like to look at the equity curve. It gives you a much better feel for the system than trying to analyze the numbers on their own.

Because of that, slippage and commissions have yet to make it into Wave59's system report. For my purposes, I already took them into account when evaluating the average trade value, and didn't really need to go farther than that. Of course, to be technically correct, they should probably be in there, and it's something we'll add soon.

One of the areas where I do spend a lot of time paying attention to slippage happens in the live account, when orders are being placed. I've already talked about this in a previous chapter, but if you can find a way to save yourself a tick here or there, that can really add up over time. The way I do it is by using limit orders rather than market orders. So instead of just buying at the ask, try entering a limit at the bid and see if you get filled there. In the ES, it's very easy to get filled like this, especially with a system that trades at the close when things tend to be a little volatile. Leave the limit on for half a minute, then go to market if you haven't been filled by then. Overall, that method has made me money, and the amount becomes noticeable the larger your size gets.

Of course, to implement something like that, you probably need to set up your system to autotrade. Once you have a mechanical system ready to go, there are two ways to actually trade it. The first way is to trade by hand, where you sit there, watch your system for signals, and then place those signals in your brokerage account yourself. This is what I've been doing for the options system I've been following. It only takes a minute or two, but you have to be available each trading day at the appropriate time. If you miss a trade, your actual results start to diverge from your theoretical results, and that tends not to work out in your favor.

The other method to trade a system is to make it fully automated. This means allowing it to actually place orders directly in your account without you needing to be there. In this case, all you have to do is make sure your computer is up and running, that your broker is connected, and that the system is active. Then, you check in around the close to see what happened. This is still not completely hands-free, as periodically you'll have

to correct for mistakes or Internet outages, but those are rare occurrences. The ES does trade through the night, so if you need to correct a trade, it can usually be done soon after the error is discovered in that market.

Both methods work, and it really just depends how much time and energy you want to spend managing your system, and how much you trust your computer to actually make financial decisions on your behalf. I'd be lying if I said my automated systems have never pulled any funny business with my account, like deciding to double-down on a particular trade. You have to be prepared to correct things when something like that happens.

For those of you interested in autotrading, the last script in the appendix provides a template for how to set that up. It will require a bit of coding for you to merge your signals into that template, but it shouldn't be too crazy. Read the notes in the script for more details on how to do this from a technical perspective.

I do have to say that it has been quite a bit of fun writing this book. It was particularly interesting to me to go back and dig some of my old systems out and apply them to recent data, and I was pleased to see how well they've all been holding up. Even William Tell, which did take a bit of a beating lately, really performed wonderfully in the 10 years since I first traded it, despite the fact that I really didn't know quite what I was doing when I originally built it. Of course, merging a bunch of patterns together is always a good idea, and you can see how the good patterns held up the bad ones over that system's lifetime. Nature figured the strength-in-numbers approach out a long time ago, and that strategy works well in markets too.

All the scripts used in this book can be found in the appendix, so you should be able to duplicate everything you've seen here. I would definitely recommend that you fool around with those scripts before trading any of them live. I used Wave59's integrated data feed almost exclusively throughout this book, but most other feeds should result in nearly identical results.

Hopefully, you'll have learned a few things about how to build systems, and might have even found a system or two worth trading on your own. I'm hopeful that some enterprising developers can take this material and transform it into something better, and (hopefully!) come back and share their results with the rest of us. There are always more rocks to look under than hands to do it with when it comes to trading and system

development in general, so I wouldn't be too surprised to see a few tweaks to this material result in some serious improvements in the performance department.

I hope you enjoyed the book, and look forward to hearing about your exploits on the bulletin board.

Happy Trading!

14. Appendix

In the following pages, you will find the source code to all of the various systems we've looked at in this manual. This is QScript® code, which requires Wave59 to run. In order to implement any of this code, you will have to type it into the QScript Editor in Wave59, and then build it, which will then make it available on your charts. Yes, you have to actually type it out by hand. I've decided to go old-school on the readers of this work, and I'm not going to include a CD that allows these scripts to be imported with the click of a mouse. I've made that decision for the following three reasons:

1) Piracy in this industry is extensive enough that trading books get copied, repackaged, and resold for pennies on the dollar. I think my *Unified Theory of Markets* manual lasted about a week before the pirates got a hold of it and started passing it around. Obviously, it takes a lot of time, energy, and resources to write a good trading book, and authors become understandably grumpy when they see all of that work get stolen. It is bad enough that many authors try publishing one book, and then stop due to the frustration of dealing with those pirates.

   I've learned a few things in my time as an author, and one of them is that if you are going to publish in this field, you have to be ready for the pirates. That's just life. I choose to publish anyway, because I value the friendships and relationships I have with the Wave59 community, so for me it's worth it. I can also say that most of the people that frequent pirate sites aren't profitable traders, and don't last very long in the industry. They also tend to be somewhat lazy. So yes, they might take the trouble to scan this book and post it somewhere, but no, they are not going to sit down and type out any code by hand. It's much easier to continue to look for the holy grail by reading one of the other hundred stolen books they just downloaded along with this one, rather than taking the time to implement and investigate these methods.

2) The second reason for forcing the readers to type these scripts is that I want to encourage people to start programming in QScript. By their nature, trading systems require coding and we need to get more of the community fluent in QScript. My hope is that after typing out and building a few working scripts,

people will see that this coding thing isn't really so bad, and will have enough experience to allow them to start fooling around on their own. QScript is probably the easiest programming language ever invented for trading, so there's no reason not to jump in and test out your own ideas. The scripts here can be used as models for how to go about doing just such a thing, so just change them a little here or there, and see what sorts of things you can find. It's fun, and you can make a lot of money too!

3) The third reason is a protection mechanism against those who probably shouldn't be trading. Yes, this was somewhat of an expensive book, at least when you compare it to the price of books found in your local bookstore. But at the same time, I've also shared scripts and methods that describe, step-by-step, how to run a small account up into the hundreds of millions of dollars. Doing something like that is possible, and has been done before by others. There's no reason anyone in this community shouldn't be able to do the same thing, if they are disciplined and able to follow their chosen system.

   Unfortunately, discipline and psychology are major factors in the success of any trader, and without them, it doesn't matter what system you choose to implement, as odds are you won't be able to stick with that method long enough to enjoy the results. So if it's really too much trouble for you to take an hour to type out some code from the back of this book, then you probably shouldn't be trading these systems. It's infinitely harder to sit through a 40-50% drawdown on your account than it is to type out some computer code, so by setting up a small hurdle here I'm actually saving some of those traders quite a bit of money, although they won't thank me for it, and will probably be the first to complain.

Finally, I understand that some of you don't understand code and will make mistakes. Don't get too worked up. All of this has been tested and works, but computers are fussy and will make you enter everything perfectly before they accept it. If you can't get it to work, check for typos. If you still have trouble, check for typos again! But if you're still stuck, of course we'll help if you open up a support ticket with your questions. I may be old-school, but I do want you to succeed if you are willing to put forth the effort.

A few quick comments. Each script starts with a header that tells you the script name, whether it is an indicator or a function, and what the scaling, etc, needs to be set to. You can enter this information into the script properties window in the QScript Editor. In the case of functions, make sure to keep the names the same as what I used, as otherwise you'll have to change them in all the scripts that end up calling them. Also, you probably want to enter all the functions first and then do all the indicators, as indicators won't build if they can't locate functions they are supposed to reference.

Any time you see a pound sign (#), that's a comment. Wave59 will ignore everything on a line after a pound sign, so you don't have to type them if you don't want to. I like to put comments in scripts so I know what the script is doing when I open it up years later.

If all of this seems totally foreign, and you are getting frustrated, go try and implement the Inverse Momentum script first. It's only a few lines long, and you'll get a chance to see how everything works without getting caught up in some of the more massive scripts. Once you get one working, the others will be much simpler.

Earik_System_WTBonds

Type: Indicator

Scaling: Price

Reference Bars: 50

Description: This implements the full William Tell Bonds system, including all 19 patterns. Make sure to run it on daily day-session only 30-year Bond futures.

```
# stopval in full points, 2.75 = $2750 on the bonds
# pattern_to_run: 0 for all, otherwise 1 through 19 for individuals
#   running at 0 isn't quite right as multiple signals are being taken at once
#   need to only take one at a time, then wait until flat for the next
# IMPORTANT: this system needs to be run on day session data only!
#      Gaps are important. If you run it on full-session data, results will be
#      questionable with this exit approach

input: stopval(2.75), pattern_to_run(0);

if (barnum==barsback) {
   entry_bar=0;
   entry_price=close;
   entry_type=0; #1=bstop, 2=blimit, 3=sstop, 4=slimit, 5=b mkt, 6=s mkt
}

# W59 doesn't always track the immediate next bar open properly, so build that in
# track whether we got filled, and exit on next profitable open if required
if (barnum==entry_bar) {
   if (entry_type == 1)  # buy stop
   {
      if (high>entry_price) {
         if (open>entry_price) entry_price=open;
         if (nextopen>entry_price) exitlong(0,close,"nextopen","onebar");
      }
   }
   if (entry_type == 2) # buy limit
   {
      if (low<entry_price) {
         if (open<entry_price) entry_price=open;
         if (nextopen>entry_price) exitlong(0,close,"nextopen","onebar");
```

```
      }
   }
   if (entry_type == 3) # sell stop
   {
      if (low<entry_price) {
         if (open<entry_price) entry_price=open;
         if (nextopen<entry_price) exitshort(0,close,"nextopen","onebar");
      }
   }
   if (entry_type == 4) # sell limit
   {
      if (high>entry_price) {
         if (open>entry_price) entry_price=open;
         if (nextopen<entry_price) exitshort(0,close,"nextopen","onebar");
      }
   }
   if (entry_type == 5) # buy market
   {
      if (nextopen>entry_price) exitlong(0,close,"nextopen","onebar");
   }
   if (entry_type == 6) # sell market
   {
      if (nextopen<entry_price) exitshort(0,close,"nextopen","onebar");
   }
}

# this will handle first profitable open exits past the immediate first bar
if (nextopen > entryprice and marketposition>0) {
   exitlong(0,close,"nextopen","onebar");
}
if (nextopen < entryprice and marketposition<0) {
   exitshort(0,close,"nextopen","onebar");
}

# this will handle stops
if (marketposition>0) exitlong(0,entryprice-stopval,"stop","onebar");
if (marketposition<0) exitshort(0,entryprice+stopval,"stop","onebar");

# compute indicators used by the patterns here
dow=dayofweek(year,month,day);
```

```
p11v1=average((h+l+c+o)/4,23);
p11v2=average((h+l+o+c)/4,5);
percr=percentr(10);

p12v1=average(h,2);
p12v2=average(l,2);
p12v3=average(c,2);
p12v4=(p12v1-p12v2)/4;
if (c>c[1]) p12v1=p12v1+(h-c);
if (c<c[1]) p12v2=p12v2-(c-l);

p13v1=average((h[1]-l[1]),10);


if (marketposition==0 or (marketposition>0 and nextopen > entryprice) or
(marketposition<0 and nextopen < entryprice))
{
  # pattern 1
  if ((pattern_to_run==1) or (pattern_to_run==0)) {
     if (dow==4 or dow==5)
     {
       if (c<o and nextopen < l) {
          buy(1,low+0.3*(h-l),"stop","onebar");
          entry_bar=barnum+1;
          entry_price=low+0.3*(h-l);
          entry_type=1; # buy stop
       }
     }
  }

  # pattern 2  ok, not very many trades -22
  if ((pattern_to_run==2) or (pattern_to_run==0)) {
     if (dow==3 or dow==4)
     {
       if (h[1]>h and l[1]<l and c>o and nextopen >=h) {
          buy(1,h-0.15*(h-l),"limit","onebar");
          entry_bar=barnum+1;
          entry_price=h-0.15*(h-l);
          entry_type=2; # buy limit
```

```
    }
  }
}

# pattern 3  not enough trades (Island Reversals)
if ((pattern_to_run==3) or (pattern_to_run==0)) {
  if (h[1]<l[2] and l>h[1] and c>=(h-.25*(h-l))) {
    buy(1,close,"nextopen","onebar");
    entry_bar=barnum+1;
    entry_price=nextopen;
    entry_type=5; #buy market
  }
}

# pattern 4 not enough trades (double island reversal)
if ((pattern_to_run==4) or (pattern_to_run==0)) {
  if (h[2]<l[3] and h[1]<l[3] and l>h[1] and l>h[2] and c>=(h-.25*(h-l))) {
     buy(1,close,"nextopen","onebar");
     entry_bar=barnum+1;
    entry_price=nextopen;
    entry_type=5; #buy market
  }
}


# pattern 5 not enough trades (island reversal short)
if ((pattern_to_run==5) or (pattern_to_run==0)) {
  if (l[1]>h[2] and h<l[1] and c<=(l+.25*(h-l))) {
    sell(1,close,"nextopen","onebar");
    entry_bar=barnum+1;
    entry_price=nextopen;
    entry_type=6; #sell market
  }
}

# pattern 6  not enough trades (double island sell)
if ((pattern_to_run==6) or (pattern_to_run==0)) {
  if (l[2]>h[3] and l[1]>h[3] and h<l[1] and h<l[2] and c<=(l+.25*(h-l))) {
    sell(1,close,"nextopen","onebar");
     entry_bar=barnum+1;
```

```
        entry_price=nextopen;
        entry_type=6; #sell market
    }
  }


  # pattern 7 good - 44 trades
  if ((pattern_to_run==7) or (pattern_to_run==0)) {
    if (dow!=1 and dow!=2)
    {
      if (l>h[1]) {
        sell(1,h[1],"stop","onebar");
        entry_bar=barnum+1;
        entry_price=h[1];
        entry_type=3; # sell stop
      }
    }
  }

  # pattern 8  bumpy towards the end
  if ((pattern_to_run==8) or (pattern_to_run==0)) {
    if (c[3]>o[3] and  h[2]<h[3] and c[2]<o[2] and h[1]<h[2] and c[1]<o[1] and l>l[1] and
c>o )
    {
        sell(1,low,"stop","onebar");
        entry_bar=barnum+1;
        entry_price=low;
        entry_type=3; # sell stop
    }
  }


  # pattern 9  hump at 1998
  if ((pattern_to_run==9) or (pattern_to_run==0)) {
    if (dow==5)
    {
      if (l > h[2] and l[1]>=h[3] and nextopen <((h-l)*.3+h)) {
        buy(1,(h+.3*(h-l)),"stop","onebar");
        entry_bar=barnum+1;
        entry_price=(h+.3*(h-l));
```

```
        entry_type=1; # buy stop
      }
   }
}

# pattern 10
if ((pattern_to_run==10) or (pattern_to_run==0)) {
   if (dow==4 or dow==5)
   {
     if (c[3]<o[3] and c[2]<o[2] and c[1]<o[1] and c>o) {
        sell(1,(c+.25*(h-l)),"limit","onebar");
        entry_bar=barnum+1;
        entry_price=(c+.25*(h-l));
        entry_type=4; # sell limit
      }
   }
}

# pattern 11
if ((pattern_to_run==11) or (pattern_to_run==0)) {
   if (percr crosses above 10 and p11v2>p11v1) {
      buy(1,high,"stop","onebar");
      entry_bar=barnum+1;
      entry_price=high;
      entry_type=1; # buy stop
   }
}

# pattern 12  good pattern, lots of trades
if ((pattern_to_run==12) or (pattern_to_run==0)) {
   if (dow==2 or dow==5)
   {
    if (p12v1[1]!=0 and p12v2[1]!=0 and p12v3[1]!=0)
    {
      if (nextopen < p12v2)  {
         buy(1,(p12v2+(p12v4*.3)),"stop","onebar");
         entry_bar=barnum+1;
         entry_price=(p12v2+(p12v4*.3));
         entry_type=1; # buy stop
       }
```

```
      }
    }
  }

  # pattern 13 can't make headway - losses bring it back to zero
  if ((pattern_to_run==13) or (pattern_to_run==0)) {
    if (dow==2 or dow==3)
    {
      if ((h[3]-l[3])>p13v1[3]) {
        if (l[3]<l[4] and l[3]<l[5] and l[3]<l[2] and l[3]<l[1] and l[3]<l and h[2]<h[1] and
h[1]<h and l[2]<l[1] and l[1]<l)
        {
          buy(1,h,"stop","onebar");
          entry_bar=barnum+1;
          entry_price=high;
          entry_type=1; # buy stop
        }
      }
    }
  }

  # pattern 14
  if ((pattern_to_run==14) or (pattern_to_run==0)) {
    if (dow==2)
    {
      if (l[1]<l[2] and l[1]<l) {
        buy(1,(h+.75*(h-l)),"stop","onebar");
        entry_bar=barnum+1;
        entry_price=h+0.75*(h-l);
        entry_type=1; # buy stop
      }
    }
  }

  # pattern 15  got hit after 2005, but up overall
  if ((pattern_to_run==15) or (pattern_to_run==0)) {
    if (dow==3)
    {
      if (c>o and c[1]>o[1] and l>=h[1]) {
        buy(1,h,"limit","onebar");
```

```
        entry_bar=barnum+1;
        entry_price=high;
        entry_type=2; # buy limit
      }
    }
}

# pattern 16  like # 10 - destruct after 98
if ((pattern_to_run==16) or (pattern_to_run==0)) {
    if (dow==4)
    {
      if (c<c[1] and(h-l)>(h[1]-l[1]))
      {
        if (nextopen>c) {
          sell(1,low,"stop","onebar");
          entry_bar=barnum+1;
          entry_price=low;
          entry_type=3; # sell stop
        }
      }
    }
}

# pattern 17  made it until 2007 or so
if ((pattern_to_run==17) or (pattern_to_run==0)) {
    if (dow==3)
    {
      if (h>h[1] and l<l[1] and c<c[1]) {
        buy(1,close,"nextopen","onebar");
        entry_bar=barnum+1;
        entry_price=nextopen;
        entry_type=5; # buy market
      }
    }
}

# pattern 18  ouch at the end
if ((pattern_to_run==18) or (pattern_to_run==0)) {
    if (dow==3)
    {
```

```
        if (h>h[1] and l<l[1] and c>c[1]) {
          sell(1,(nextopen -0.2*(h-l)),"stop","onebar");
          entry_bar=barnum+1;
          entry_price=nextopen -0.2*(h-l);
          entry_type=3; # sell stop
        }
      }
    }

    # pattern 19 sideways - this seasonal seems to have gone away
    if ((pattern_to_run==19) or (pattern_to_run==0)) {
      if (month==5 and dow==1) {
        buy(1,close,"nextopen","onebar");
        entry_bar=barnum+1;
        entry_price=nextopen;
        entry_type=5; # buy market
      }
    }

}
```

Earik_System_AdaptiveCrossoverFade

Type: Indicator

Scaling: Price

Reference Bars: 250

Description: This implements an adaptive system to find the optimal moving average periods, but then fade them. Enter the Earik_SimpleAvgCrossover function first before typing out this script.

```
# lookback is how many bars to use when computing equity curves
# for the adaptive system
input:lookback(100);

if (barnum==barsback) {
   bs=0;
   longshort=0;
   temp=0;
}

# this section runs a bunch of moving average combinations
# the best combination's signal is saved, and then trades
# are placed in the OPPOSITE direction - we are trading
# the market's tendency to avoid stable parameter sets here

equity=Earik_SimpleAvgCrossover(10,20,lookback,&longshort);
longshort*=-1; #we're doing the opposite...

test=Earik_SimpleAvgCrossover(10,30,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,40,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,50,lookback,&temp);
```

```
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,60,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,70,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,80,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,90,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(10,100,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,30,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}
```

```
test=Earik_SimpleAvgCrossover(20,40,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,50,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,60,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,70,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,80,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,90,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(20,100,lookback,&temp);
if (test>equity) {
   equity=test;
```

```
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,40,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,50,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,60,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,70,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,80,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,90,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(30,100,lookback,&temp);
```

```
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(40,50,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(40,60,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(40,70,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(40,80,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(40,90,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(40,100,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}
```

```
test=Earik_SimpleAvgCrossover(50,60,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(50,70,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(50,80,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(50,90,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}

test=Earik_SimpleAvgCrossover(50,100,lookback,&temp);
if (test>equity) {
   equity=test;
   longshort=-1*temp;
}


# now make sure we are going against the direction of the best
# selection...

if (longshort!=bs) {
   if (longshort>0) {
      buy(1,close,"market","onebar");
      bs=1;
   }
```

```
    if (longshort<0) {
        sell(1,close,"market","onebar");
        bs=-1;
    }
}
```

Earik_SimpleAvgCrossover

Type: Function

Reference Bars: 100

Description: This is a helper function used in the Earik_System_AdaptiveCrossoverFade system. Program it first, then do the indicator.

```
# this computes the equity over 'lookback' bars when trading
#    a simple moving average crossover
# if longshort is called as a reference variable (like &longshort)
#     then it will store whether this system is currently long (1) or short (-1)
input: short_length long_length, lookback, longshort;

if (barnum==barsback) {
   profit=0;
   bs=0;
}

avg1=average(close,short_length);
avg2=average(close,long_length);

# compute equity through this bar
if (bs>0) profit+=close-close[1];
if (bs<0) profit+=close[1]-close;

# buy signal
if (avg1 crosses over avg2) {
   bs=1;
}

# sell signal
if (avg1 crosses under avg2) {
   bs=-1;
}

# return values
longshort=bs;
return profit-profit[lookback];
```

Earik_System_BCI

Type: Indicator
Scaling: Price
Reference Bars: 15
Description: BCI Reversal System, trading at the next open.

```
if (barnum==barsback)
  bs=0;

trigger=sqrt(h*l);
if (nextopen<=trigger) {
   if (bs<=0) {
      buy(1,nextopen,"nextopen","onebar");
      bs=1;
   }
}
else {
   if (bs>=0) {
      sell(1,nextopen,"nextopen","onebar");
      bs=-1;
   }
}
```

Earik_System_BookSphericalTester

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: Use this script to poke around and find interesting patterns using Spherical Astrology.

```
# p1 is first planet (0 through 10), p2 is second planet
# trigger is what ratio to use for a signal (range is 0-180)
# hold is how many bars to hold for after an entry

input: p1(1), p2(1), trigger(5), hold(15);
if (barnum==barsback) {
   bs=0;
   entrybar=barsback;
}

if (barnum==barsback) {
   if (p1==0) plan1=astro_moon;
   if (p1==1) plan1=astro_mercury;
   if (p1==2) plan1=astro_venus;
   if (p1==3) plan1=astro_sun;
   if (p1==4) plan1=astro_mars;
   if (p1==5) plan1=astro_jupiter;
   if (p1==6) plan1=astro_saturn;
   if (p1==7) plan1=astro_uranus;
   if (p1==8) plan1=astro_neptune;
   if (p1==9) plan1=astro_pluto;
   if (p1==10) plan1=astro_true_node;

   if (p2==0) plan2=astro_moon;
   if (p2==1) plan2=astro_mercury;
   if (p2==2) plan2=astro_venus;
   if (p2==3) plan2=astro_sun;
   if (p2==4) plan2=astro_mars;
   if (p2==5) plan2=astro_jupiter;
   if (p2==6) plan2=astro_saturn;
   if (p2==7) plan2=astro_uranus;
   if (p2==8) plan2=astro_neptune;
   if (p2==9) plan2=astro_pluto;
```

```
    if (p2==10) plan2=astro_true_node;
}

if (p1<p2) {
    long1= astro(year,month,day,time,plan1,true,astro_long,true);
    dec1= astro_declination(year,month,day,time,plan1,true);

    long2= astro(year,month,day,time,plan2,true,astro_long,true);
    dec2= astro_declination(year,month,day,time,plan2,true);

    # Compute Aspect or Ratio Values in the range 0-180
    aspect=astro_xyz_ratio(long1,dec1,long2,dec2)*180;

    plot1=aspect;
    plot2=trigger;

    if (aspect crosses over trigger) {
    # if (aspect crosses under trigger) { # uncomment for shorts
        buy(1,close,"market","onebar");
        # sell(1,close,"market","onebar"); # uncomment for shorts
        entrybar=barnum;
        bs=1;
    }
    if ((bs>0) and (barnum>=entrybar+hold))
    {
        bs=0;
        exitlong(0,close,"market","onebar");
        exitshort(0,close,"market","onebar");
    }
}
```

PUC_Lunar_Reversal

Type: Indicator
Scaling: Screen
Reference Bars: 15
Description: This is the Lunar Reversal system (Moon-Sun) originally presented at the 2011 PowerUser Conference in Las Vegas.

```
# this script implements a reversal system based on the Lunar Key
# buy_level = what degree to buy at
# sell_level1, sell_level2 = what degrees to sell at
# use_spherical - if true, use spherical ratios, otherwise use traditional aspects
# short_filter - if true, only take short trades on an up close

# Copyright 2011 by Wave59 Technologies Int'l Inc.
# All rights reserved. Do not make illegal copies of this material.

input: buy_level(112), sell_level1(5), sell_level2(175), use_spherical(true),
short_filter(true);

if (barnum==barsback) {
   entrybar=0;
   holding_period=7;
   bs=0;
}

# Compute Moon and Sun values
long1= astro(year,month,day,time,astro_moon,true,astro_long,true);
dec1= astro_declination(year,month,day,time,astro_moon,true);

long2= astro(year,month,day,time,astro_sun,true,astro_long,true);
dec2= astro_declination(year,month,day,time,astro_sun,true);

# Compute Aspect or Ratio Values in the range 0-180
if (use_spherical==true) {
   aspect=astro_xyz_ratio(long1,dec1,long2,dec2)*180;
}
else {
   aspect=long1-long2;
   if (aspect<0) aspect+=360;
   if (aspect>180) aspect=360-aspect;
```

```
}

# Implement system
if ((aspect crosses over sell_level1) or (aspect crosses under sell_level2))
# if (go_short==true )
{
   bs=-1;
   entrybar=barnum;
}

if ((aspect crosses over buy_level) or (aspect crosses under buy_level))
{
   bs=1;
   entrybar=barnum;
}

# buy signals
if (bs==1) {
   if ((true) and (marketposition<=0))
   {
      cancelall();
      buy(1,close,"market","onebar");
      bs=0;
   }
}

# sell signals
if (bs==-1) {
   # if using short filter, check for up close
   if (short_filter) {
      if (c>c[1]) filter=true;
      else filter=false;
   }
   else filter=true;

   if ((filter==true) and (marketposition>=0))
   {
      cancelall();
      sell(1,close,"market","onebar");
      bs=0;
```

```
    }
}

# plot everything
plot1=aspect;
plot2=sell_level1;
plot3=sell_level2;
plot4=buy_level;
style2, style3, style4=ps_dot;
color2, color3=fuchsia;
color4=green;
```

PUC_LunarSignal

Type: Function

Reference Bars: 15

Description: This is the lunar reversal system, but in function format. This is used later by the merged voting system.

```
# this script implements a reversal system based on the Lunar Key
# buy_level = what degree to buy at
# sell_level1, sell_level2 = what degrees to sell at
# use_spherical - if true, use spherical ratios, otherwise use traditional aspects
# short_filter - if true, only take short trades on an up close

# Copyright 2011 by Wave59 Technologies Int'l Inc.
# All rights reserved. Do not make illegal copies of this material.

input:buy_level,sell_level1,sell_level2,use_spherical,short_filter;

if (barnum==barsback) {
   entrybar=0;
   holding_period=7;
   bs=0;
   longshort=0;
}

# Compute Moon and Sun values
long1= astro(year,month,day,time,astro_moon,true,astro_long,true);
dec1= astro_declination(year,month,day,time,astro_moon,true);

long2= astro(year,month,day,time,astro_sun,true,astro_long,true);
dec2= astro_declination(year,month,day,time,astro_sun,true);

# Compute Aspect or Ratio Values in the range 0-180
if (use_spherical==true) {
   aspect=astro_xyz_ratio(long1,dec1,long2,dec2)*180;
}
else {
   aspect=long1-long2;
   if (aspect<0) aspect+=360;
   if (aspect>180) aspect=360-aspect;
}
```

```
# Implement system
if ((aspect crosses over sell_level1) or (aspect crosses under sell_level2)) {
   bs=-1;
   entrybar=barnum;
}

if ((aspect crosses over buy_level) or (aspect crosses under buy_level)) {
   bs=1;
   entrybar=barnum;
}

# buy signals
if (bs==1) {
   if ((true) and (marketposition<=0)) {
      cancelall();
      longshort=1;
      bs=0;
   }
}

# sell signals
if (bs==-1) {
   # if using short filter, check for up close
   if (short_filter) {
      if (c>c[1]) filter=true;
      else filter=false;
   }
   else filter=true;

   if ((filter==true) and (marketposition>=0)) {
      cancelall();
      longshort=-1;
      bs=0;
   }
}

return longshort;
```

PUC_Lunar_NthProfitableClose

Type: Indicator
Scaling: Screen
Reference Bars: 15
Description: This is the Lunar System using an Nth-profitable close exit method. Originally presented at the 2011 PowerUser Conference in Las Vegas.

```
# this script implements a system based on the Lunar Key
# trades are entered based on Lunar Key, and exited at the Nth profitable close
# nth_proitable_close = which close to exit at? 1 = 1st profitable close, 2 = 2nd, etc
# buy_level = what degree to buy at
# sell_level1, sell_level2 = what degrees to sell at
# use_spherical - if true, use spherical ratios, otherwise use traditional aspects
# short_filter - if true, only take short trades on an up close

# Copyright 2011 by Wave59 Technologies Int'l Inc.
# All rights reserved. Do not make illegal copies of this material.

input: nth_profitable_close(1), buy_level(112), sell_level1(5), sell_level2(175),
use_spherical(true), short_filter(true);

# initialize variables
if (barnum==barsback) {
   entrybar=0;
   bs=0;
   myentry=close;
   closecount=0;
}

# Compute Moon and Sun values
long1= astro(year,month,day,time,astro_moon,true,astro_long,true);
dec1= astro_declination(year,month,day,time,astro_moon,true);

long2= astro(year,month,day,time,astro_sun,true,astro_long,true);
dec2= astro_declination(year,month,day,time,astro_sun,true);

# Compute Aspect or Ratio Values in the range 0-180
if (use_spherical==true) {
   aspect=astro_xyz_ratio(long1,dec1,long2,dec2)*180;
}
```

```
else {
   aspect=long1-long2;
   if (aspect<0) aspect+=360;
   if (aspect>180) aspect=360-aspect;
}


# Implement system
if ((aspect crosses over sell_level1) or (aspect crosses under sell_level2))
{
   bs=-1;
   entrybar=barnum;
}

if ((aspect crosses over buy_level) or (aspect crosses under buy_level))
{
   bs=1;
   entrybar=barnum;

}

# longs
if (bs==1) {
   if ((true) and (marketposition<=0))
   {
      cancelall();
      buy(1,close,"market","onebar");
      bs=0;
      myentry=close;
      closecount=1;
   }
}

# shorts
if (bs==-1) {

   # if using short filter, check for up close
   if (short_filter) {
      if (c>c[1]) filter=true;
      else filter=false;
```

```
    }
    else filter=true;

    if ((filter==true) and (marketposition>=0))
    {
        cancelall();
        sell(1,close,"market","onebar");
        bs=0;
        myentry=close;
        closecount=1;
    }
}

# exit one on Nth profitable close
# use "closecount" variable to keep track
if ((close>myentry) and (marketposition>0))
{
    if (closecount==nth_profitable_close)
        exitlong(1,close,"market","onebar");
    else
        closecount+=1;

}
if ((close<myentry) and (marketposition<0))
{
    if (closecount==nth_profitable_close)
        exitshort(1,close,"market","onebar");
     else
        closecount+=1;
}

# plot everything
plot1=aspect;
plot2=sell_level1;
plot3=sell_level2;
plot4=buy_level;
style2,style3,style4=ps_dot;
color2,color3=fuchsia;
color4=green;
```

PUC_Lunar_MultipleCt_ProfitableCloseExit

Type: Indicator
Scaling: Screen
Reference Bars: 15
Description: This is the Lunar System using an Nth profitable close exit method, trading more than one contract together. Originally presented at the 2011 PowerUser Conference in Las Vegas.

```
# this script implements a system based on the Lunar Key
# X contracts are entered based on Lunar Key, and one contract is peeled off at each
profitable close

# initial_contracts = how many contracts to open a position with
# buy_level = what degree to buy at
# sell_level1, sell_level2 = what degrees to sell at
# use_spherical - if true, use spherical ratios, otherwise use traditional aspects
# short_filter - if true, only take short trades on an up close

# Copyright 2011 by Wave59 Technologies Int'l Inc.
# All rights reserved. Do not make illegal copies of this material.

input: initial_contracts(3), buy_level(112), sell_level1(5), sell_level2(175),
use_spherical(true), short_filter(true);

# initialize variables
if (barnum==barsback) {
   entrybar=0;
   bs=0;
   myentry=close;
}

# Compute Moon and Sun values
long1= astro(year,month,day,time,astro_moon,true,astro_long,true);
dec1= astro_declination(year,month,day,time,astro_moon,true);

long2= astro(year,month,day,time,astro_sun,true,astro_long,true);
dec2= astro_declination(year,month,day,time,astro_sun,true);

# Compute Aspect or Ratio Values in the range 0-180
if (use_spherical==true) {
```

```
   aspect=astro_xyz_ratio(long1,dec1,long2,dec2)*180;
}
else {
   aspect=long1-long2;
   if (aspect<0) aspect+=360;
   if (aspect>180) aspect=360-aspect;
}


# system
if ((aspect crosses over sell_level1) or (aspect crosses under sell_level2))
{
   bs=-1;
   entrybar=barnum;
}

if ((aspect crosses over buy_level) or (aspect crosses under buy_level))
{
   bs=1;
   entrybar=barnum;
}

# longs
if (bs==1) {
   if ((true) and (marketposition<=0))
   {
      cancelall();
      buy(initial_contracts,close,"market","onebar");
      bs=0;
      myentry=close;
   }
}

# shorts
if (bs==-1) {
   # if using short filter, check for up close
   if (short_filter) {
      if (c>c[1]) filter=true;
      else filter=false;
   }
```

```
   else filter=true;

   if ((filter==true) and (marketposition>=0))
   {
      cancelall();
      sell(initial_contracts,close,"market","onebar");
      bs=0;
      myentry=close;
   }
}

# exit one on a profitable close
if (close>myentry)
{
   exitlong(1,close,"market","onebar");
}
if (close<myentry)
{
   exitshort(1,close,"market","onebar");
}

# plot everything
plot1=aspect;
plot2=sell_level1;
plot3=sell_level2;
plot4=buy_level;
style2,style3,style4=ps_dot;
color2,color3=fuchsia;
color4=green;
```

Hive_Chicago

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: This is the Chicago Hive reversal system. Note that it is an indicator, with an identically named function that actually computes the signal. You will need both.

```
# This script has been machine generated using the Hive Designer
# It is a wrapper for the core hive function 'hive_chicago'

input: signal_color(red), threshold_color(blue), thickness(1);

# run the hive function
signal = hive_chicago();

# plot signals on chart
plot1 = signal;
color1 = signal_color;
plot2 = 0;
color2 = threshold_color;
style2 = ps_dot;
thickness1, thickness2 = thickness;

# implement trading system
if ((signal>0) and (marketposition<=0))
  buy(1,close,"market","onebar");
if ((signal<0) and (marketposition>=0))
  sell(1,close,"market","onebar");
```

Hive_Chicago

Type: Function
Reference Bars: 50
Description: This is the function that actually implements the hive_chicago calculation, and has the same name as the indicator. This one might be a bear to type. If you have an active W59 subscription, you can download this for free from the QScript library at www.wave59.com/library

---

```
# This script has been machine generated using the Hive Designer
# Changing anything may result in your Hive (and W59) crashing!

# Initialization Block
if (barnum == barsback) {
  ref=hive_init(147);
  i0,i1,i2,i3,i4,i5,i6,i7,i8,i9 = 0;
  i10,i11,i12,i13,i14,i15,i16,i17,i18,i19 = 0;
  i20,i21,i22,i23,i24,i25,i26,i27,i28,i29 = 0;
  i30,i31,i32,i33,i34,i35,i36,i37,i38,i39 = 0;
  i40,i41,i42,i43,i44,i45,i46,i47,i48,i49 = 0;
  i50,i51,i52,i53,i54,i55,i56,i57,i58,i59 = 0;
  i60,i61,i62,i63,i64,i65,i66,i67,i68,i69 = 0;
  i70,i71,i72,i73,i74,i75,i76,i77,i78,i79 = 0;
  i80,i81,i82,i83,i84,i85,i86,i87,i88,i89 = 0;
  i90,i91,i92,i93,i94,i95,i96,i97,i98,i99 = 0;
  i100,i101,i102,i103,i104,i105,i106,i107,i108,i109 = 0;
  i110,i111,i112,i113,i114,i115,i116,i117,i118,i119 = 0;
  i120,i121,i122,i123,i124,i125,i126,i127,i128,i129 = 0;
  i130,i131,i132,i133,i134,i135,i136,i137,i138,i139 = 0;
  i140,i141,i142,i143,i144,i145,i146 = 0;
}

# Compute All Possible Inputs
in0 = AMA(c,2,0.5,0.65,0.8);
hive_unbounded_to_signals(in0,&i0,&i1,&i2);
in1 = AMA(c,3,0.5,0.65,0.8);
hive_unbounded_to_signals(in1,&i3,&i4,&i5);
in2 = AMA(c,4,0.5,0.65,0.8);
hive_unbounded_to_signals(in2,&i6,&i7,&i8);
in3 = AMA(c,5,0.5,0.65,0.8);
```

```
hive_unbounded_to_signals(in3,&i9,&i10,&i11);
in4 = AMA(c,6,0.5,0.65,0.8);
hive_unbounded_to_signals(in4,&i12,&i13,&i14);
in5 = AMA(c,7,0.5,0.65,0.8);
hive_unbounded_to_signals(in5,&i15,&i16,&i17);
in6 = AMA(c,8,0.5,0.65,0.8);
hive_unbounded_to_signals(in6,&i18,&i19,&i20);
in7 = AMA(c,9,0.5,0.65,0.8);
hive_unbounded_to_signals(in7,&i21,&i22,&i23);
in8 = AMA(c,10,0.5,0.65,0.8);
hive_unbounded_to_signals(in8,&i24,&i25,&i26);
in9 = AMA(c,11,0.5,0.65,0.8);
hive_unbounded_to_signals(in9,&i27,&i28,&i29);
in10 = AMA(c,12,0.5,0.65,0.8);
hive_unbounded_to_signals(in10,&i30,&i31,&i32);
in11 = AMA(c,13,0.5,0.65,0.8);
hive_unbounded_to_signals(in11,&i33,&i34,&i35);
in12 = AMA(c,14,0.5,0.65,0.8);
hive_unbounded_to_signals(in12,&i36,&i37,&i38);
in13 = AMA(c,15,0.5,0.65,0.8);
hive_unbounded_to_signals(in13,&i39,&i40,&i41);
in14 = AMA(c,16,0.5,0.65,0.8);
hive_unbounded_to_signals(in14,&i42,&i43,&i44);
in15 = AMA(c,17,0.5,0.65,0.8);
hive_unbounded_to_signals(in15,&i45,&i46,&i47);
in16 = AMA(c,18,0.5,0.65,0.8);
hive_unbounded_to_signals(in16,&i48,&i49,&i50);
in17 = AMA(c,19,0.5,0.65,0.8);
hive_unbounded_to_signals(in17,&i51,&i52,&i53);
in18 = AMA(c,20,0.5,0.65,0.8);
hive_unbounded_to_signals(in18,&i54,&i55,&i56);
in19 = AMA(c,21,0.5,0.65,0.8);
hive_unbounded_to_signals(in19,&i57,&i58,&i59);
in20 = AMA(c,22,0.5,0.65,0.8);
hive_unbounded_to_signals(in20,&i60,&i61,&i62);
in21 = AMA(c,23,0.5,0.65,0.8);
hive_unbounded_to_signals(in21,&i63,&i64,&i65);
in22 = AMA(c,24,0.5,0.65,0.8);
hive_unbounded_to_signals(in22,&i66,&i67,&i68);
in23 = AMA(c,25,0.5,0.65,0.8);
```

```
hive_unbounded_to_signals(in23,&i69,&i70,&i71);
in24 = AMA(c,26,0.5,0.65,0.8);
hive_unbounded_to_signals(in24,&i72,&i73,&i74);
in25 = AMA(c,27,0.5,0.65,0.8);
hive_unbounded_to_signals(in25,&i75,&i76,&i77);
in26 = AMA(c,28,0.5,0.65,0.8);
hive_unbounded_to_signals(in26,&i78,&i79,&i80);
in27 = AMA(c,29,0.5,0.65,0.8);
hive_unbounded_to_signals(in27,&i81,&i82,&i83);
in28 = AMA(c,30,0.5,0.65,0.8);
hive_unbounded_to_signals(in28,&i84,&i85,&i86);
in29 = AMA(c,31,0.5,0.65,0.8);
hive_unbounded_to_signals(in29,&i87,&i88,&i89);
in30 = AMA(c,32,0.5,0.65,0.8);
hive_unbounded_to_signals(in30,&i90,&i91,&i92);
in31 = AMA(c,33,0.5,0.65,0.8);
hive_unbounded_to_signals(in31,&i93,&i94,&i95);
in32 = AMA(c,34,0.5,0.65,0.8);
hive_unbounded_to_signals(in32,&i96,&i97,&i98);
in33 = AMA(c,35,0.5,0.65,0.8);
hive_unbounded_to_signals(in33,&i99,&i100,&i101);
in34 = AMA(c,36,0.5,0.65,0.8);
hive_unbounded_to_signals(in34,&i102,&i103,&i104);
in35 = AMA(c,37,0.5,0.65,0.8);
hive_unbounded_to_signals(in35,&i105,&i106,&i107);
in36 = AMA(c,38,0.5,0.65,0.8);
hive_unbounded_to_signals(in36,&i108,&i109,&i110);
in37 = AMA(c,39,0.5,0.65,0.8);
hive_unbounded_to_signals(in37,&i111,&i112,&i113);
in38 = AMA(c,40,0.5,0.65,0.8);
hive_unbounded_to_signals(in38,&i114,&i115,&i116);
in39 = AMA(c,41,0.5,0.65,0.8);
hive_unbounded_to_signals(in39,&i117,&i118,&i119);
in40 = AMA(c,42,0.5,0.65,0.8);
hive_unbounded_to_signals(in40,&i120,&i121,&i122);
in41 = AMA(c,43,0.5,0.65,0.8);
hive_unbounded_to_signals(in41,&i123,&i124,&i125);
in42 = AMA(c,44,0.5,0.65,0.8);
hive_unbounded_to_signals(in42,&i126,&i127,&i128);
in43 = AMA(c,45,0.5,0.65,0.8);
```

```
hive_unbounded_to_signals(in43,&i129,&i130,&i131);
in44 = AMA(c,46,0.5,0.65,0.8);
hive_unbounded_to_signals(in44,&i132,&i133,&i134);
in45 = AMA(c,47,0.5,0.65,0.8);
hive_unbounded_to_signals(in45,&i135,&i136,&i137);
in46 = AMA(c,48,0.5,0.65,0.8);
hive_unbounded_to_signals(in46,&i138,&i139,&i140);
in47 = AMA(c,49,0.5,0.65,0.8);
hive_unbounded_to_signals(in47,&i141,&i142,&i143);
in48 = AMA(c,50,0.5,0.65,0.8);
hive_unbounded_to_signals(in48,&i144,&i145,&i146);

# Update Data
hive_updatedata(ref,barnum,close);
hive_updateinput(ref,barnum,0,i0);
hive_updateinput(ref,barnum,1,i1);
hive_updateinput(ref,barnum,2,i2);
hive_updateinput(ref,barnum,3,i3);
hive_updateinput(ref,barnum,4,i4);
hive_updateinput(ref,barnum,5,i5);
hive_updateinput(ref,barnum,6,i6);
hive_updateinput(ref,barnum,7,i7);
hive_updateinput(ref,barnum,8,i8);
hive_updateinput(ref,barnum,9,i9);
hive_updateinput(ref,barnum,10,i10);
hive_updateinput(ref,barnum,11,i11);
hive_updateinput(ref,barnum,12,i12);
hive_updateinput(ref,barnum,13,i13);
hive_updateinput(ref,barnum,14,i14);
hive_updateinput(ref,barnum,15,i15);
hive_updateinput(ref,barnum,16,i16);
hive_updateinput(ref,barnum,17,i17);
hive_updateinput(ref,barnum,18,i18);
hive_updateinput(ref,barnum,19,i19);
hive_updateinput(ref,barnum,20,i20);
hive_updateinput(ref,barnum,21,i21);
hive_updateinput(ref,barnum,22,i22);
hive_updateinput(ref,barnum,23,i23);
hive_updateinput(ref,barnum,24,i24);
hive_updateinput(ref,barnum,25,i25);
```

```
hive_updateinput(ref,barnum,26,i26);
hive_updateinput(ref,barnum,27,i27);
hive_updateinput(ref,barnum,28,i28);
hive_updateinput(ref,barnum,29,i29);
hive_updateinput(ref,barnum,30,i30);
hive_updateinput(ref,barnum,31,i31);
hive_updateinput(ref,barnum,32,i32);
hive_updateinput(ref,barnum,33,i33);
hive_updateinput(ref,barnum,34,i34);
hive_updateinput(ref,barnum,35,i35);
hive_updateinput(ref,barnum,36,i36);
hive_updateinput(ref,barnum,37,i37);
hive_updateinput(ref,barnum,38,i38);
hive_updateinput(ref,barnum,39,i39);
hive_updateinput(ref,barnum,40,i40);
hive_updateinput(ref,barnum,41,i41);
hive_updateinput(ref,barnum,42,i42);
hive_updateinput(ref,barnum,43,i43);
hive_updateinput(ref,barnum,44,i44);
hive_updateinput(ref,barnum,45,i45);
hive_updateinput(ref,barnum,46,i46);
hive_updateinput(ref,barnum,47,i47);
hive_updateinput(ref,barnum,48,i48);
hive_updateinput(ref,barnum,49,i49);
hive_updateinput(ref,barnum,50,i50);
hive_updateinput(ref,barnum,51,i51);
hive_updateinput(ref,barnum,52,i52);
hive_updateinput(ref,barnum,53,i53);
hive_updateinput(ref,barnum,54,i54);
hive_updateinput(ref,barnum,55,i55);
hive_updateinput(ref,barnum,56,i56);
hive_updateinput(ref,barnum,57,i57);
hive_updateinput(ref,barnum,58,i58);
hive_updateinput(ref,barnum,59,i59);
hive_updateinput(ref,barnum,60,i60);
hive_updateinput(ref,barnum,61,i61);
hive_updateinput(ref,barnum,62,i62);
hive_updateinput(ref,barnum,63,i63);
hive_updateinput(ref,barnum,64,i64);
hive_updateinput(ref,barnum,65,i65);
```

```
hive_updateinput(ref,barnum,66,i66);
hive_updateinput(ref,barnum,67,i67);
hive_updateinput(ref,barnum,68,i68);
hive_updateinput(ref,barnum,69,i69);
hive_updateinput(ref,barnum,70,i70);
hive_updateinput(ref,barnum,71,i71);
hive_updateinput(ref,barnum,72,i72);
hive_updateinput(ref,barnum,73,i73);
hive_updateinput(ref,barnum,74,i74);
hive_updateinput(ref,barnum,75,i75);
hive_updateinput(ref,barnum,76,i76);
hive_updateinput(ref,barnum,77,i77);
hive_updateinput(ref,barnum,78,i78);
hive_updateinput(ref,barnum,79,i79);
hive_updateinput(ref,barnum,80,i80);
hive_updateinput(ref,barnum,81,i81);
hive_updateinput(ref,barnum,82,i82);
hive_updateinput(ref,barnum,83,i83);
hive_updateinput(ref,barnum,84,i84);
hive_updateinput(ref,barnum,85,i85);
hive_updateinput(ref,barnum,86,i86);
hive_updateinput(ref,barnum,87,i87);
hive_updateinput(ref,barnum,88,i88);
hive_updateinput(ref,barnum,89,i89);
hive_updateinput(ref,barnum,90,i90);
hive_updateinput(ref,barnum,91,i91);
hive_updateinput(ref,barnum,92,i92);
hive_updateinput(ref,barnum,93,i93);
hive_updateinput(ref,barnum,94,i94);
hive_updateinput(ref,barnum,95,i95);
hive_updateinput(ref,barnum,96,i96);
hive_updateinput(ref,barnum,97,i97);
hive_updateinput(ref,barnum,98,i98);
hive_updateinput(ref,barnum,99,i99);
hive_updateinput(ref,barnum,100,i100);
hive_updateinput(ref,barnum,101,i101);
hive_updateinput(ref,barnum,102,i102);
hive_updateinput(ref,barnum,103,i103);
hive_updateinput(ref,barnum,104,i104);
hive_updateinput(ref,barnum,105,i105);
```

```
hive_updateinput(ref,barnum,106,i106);
hive_updateinput(ref,barnum,107,i107);
hive_updateinput(ref,barnum,108,i108);
hive_updateinput(ref,barnum,109,i109);
hive_updateinput(ref,barnum,110,i110);
hive_updateinput(ref,barnum,111,i111);
hive_updateinput(ref,barnum,112,i112);
hive_updateinput(ref,barnum,113,i113);
hive_updateinput(ref,barnum,114,i114);
hive_updateinput(ref,barnum,115,i115);
hive_updateinput(ref,barnum,116,i116);
hive_updateinput(ref,barnum,117,i117);
hive_updateinput(ref,barnum,118,i118);
hive_updateinput(ref,barnum,119,i119);
hive_updateinput(ref,barnum,120,i120);
hive_updateinput(ref,barnum,121,i121);
hive_updateinput(ref,barnum,122,i122);
hive_updateinput(ref,barnum,123,i123);
hive_updateinput(ref,barnum,124,i124);
hive_updateinput(ref,barnum,125,i125);
hive_updateinput(ref,barnum,126,i126);
hive_updateinput(ref,barnum,127,i127);
hive_updateinput(ref,barnum,128,i128);
hive_updateinput(ref,barnum,129,i129);
hive_updateinput(ref,barnum,130,i130);
hive_updateinput(ref,barnum,131,i131);
hive_updateinput(ref,barnum,132,i132);
hive_updateinput(ref,barnum,133,i133);
hive_updateinput(ref,barnum,134,i134);
hive_updateinput(ref,barnum,135,i135);
hive_updateinput(ref,barnum,136,i136);
hive_updateinput(ref,barnum,137,i137);
hive_updateinput(ref,barnum,138,i138);
hive_updateinput(ref,barnum,139,i139);
hive_updateinput(ref,barnum,140,i140);
hive_updateinput(ref,barnum,141,i141);
hive_updateinput(ref,barnum,142,i142);
hive_updateinput(ref,barnum,143,i143);
hive_updateinput(ref,barnum,144,i144);
hive_updateinput(ref,barnum,145,i145);
```

```
hive_updateinput(ref,barnum,146,i146);

# Compute Signal
signal = hive_execute(ref,barnum);

return signal;
```

Hive_Chicago2

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: This is the Chicago Hive system with filters using the relationship between the current bar's close and the close two bars ago.

```
input: signal_color(red), threshold_color(blue), thickness(1);

# run the hive function
signal = hive_chicago();

# plot signals on chart
plot1 = signal;
color1 = signal_color;
plot2 = 0;
color2 = threshold_color;
style2 = ps_dot;
thickness1, thickness2 = thickness;

# if close<close[2] and hive is bullish, go long
if (signal>0) {
   if ((marketposition<0) or ((c<c[2]) and (marketposition==0)))
      buy(1,close,"market","onebar");
   if ((marketposition>0) and (c>c[2]))
      exitlong(0,close,"market","onebar");
}
# if close>close[2] and hive is bearish, go short
if (signal<0) {
   if ((marketposition>0) or ((c>c[2]) and (marketposition==0)))
      sell(1,close,"market","onebar");
   if ((marketposition<0) and (c<c[2]))
      exitshort(0,close,"market","onebar");
}
```

Earik_System_UltramomInv

Type: Indicator
Scaling: Screen
Reference Bars: 15
Description: This system trades zero crossovers by the USM curve, going the opposite way as expected. Originally discovered by a genetic algorithm.

```
input: len(8);
mom = ultramom(c,len,4);
if (mom crosses over 0) sell(1,close,"market","onebar");
if (mom crosses under 0) buy(1,close,"market","onebar");

plot1=mom;
color1=blue;
plot2=0;
```

Earik_System_BookGA1

Type: Indicator
Scaling: Screen
Reference Bars: 40
Description: This system was developed by a genetic algorithm, and was the first such system to be discussed in this book.

```
if (barnum==barsback)
   bs=0;

# compute the signal ----------------------------------------------------
signal=0;

# usm calculation ...
denom1 = close[1]-open[1];
denom2 = close-close[1];
if ((denom1!=0) and (denom2!=0)) {
   signal=(((close-high[1])/denom1) + (low[1]-low[2]))/denom2;
} else
   signal=0;
signal -= ultramom(close,8,4);
signal = sin(signal);

# weird bar pattern filter ...
if ((low[2]-low[3]==1) or (high[2]-high[3]==1))
   signal = close-high[1];

# trading system -------------------------------------------------------
if ((signal>0) and (bs<=0)) {
   buy(1,close,"market","onebar");
   bs=1;
}
if ((signal<0) and (bs>=0)) {
   sell(1,close,"market","onebar");
   bs=-1;
}

plot1=sign(signal);
plot2=0;
color1=green;
```

Earik_System_BookGA2

Type: Indicator
Scaling: Screen
Reference Bars: 40
Description: This is the second system we looked at that was developed by a genetic algorithm.

```
if (barnum==barsback)
   bs=0;

signal=0;

# compute the signal ------------------------------------------------
usm2 = ultramom(close,2,4);
usm10 = ultramom(close,10,4);
usm18 = ultramom(close,18,4);

signal=0;
if (usm18!=0 and usm2!=0) {
   signal = (1/usm18)*(1/usm18) - (1/usm2);
   sine_val = sin(usm10);
   if (signal > sine_val) signal = 1; else if (signal<sine_val) signal = -1;
}

# trading system ----------------------------------------------------
if ((signal>0) and (bs<=0)) {
   buy(1,close,"market","onebar");
   bs=1;
}
if ((signal<0) and (bs>=0)) {
   sell(1,close,"market","onebar");
   bs=-1;
}

plot1=sign(signal);
plot2=0;
color1=green;
```

Earik_System_RandomTest

Type: Indicator
Scaling: Screen
Reference Bars: 20
Description: This system trades using random numbers.

```
# optimize: use this to run the system multiple times through the optimizer. It has
#    no effect aside from resetting the random number generator
# tradebars: place a trade every 'tradebar' bars
# goodbars: exit after this many profitable closes
# badbars: exit after this many unprofitable closes

input: optimize(1), tradebars(20), goodbars(2), badbars(6);
if (barnum==barsback) {
   bs=0;
   entry=close;
   good=0;
   bad=0;
}

if (bs==1) {
   if (close>entry) {
      good+=1;
      if (good>=goodbars) {
         exitlong(0,close,"market","onebar");
         bs=0;
      }
   }
   else if (close<entry) {
      bad+=1;
      if (bad>=badbars) {
         exitlong(0,close,"market","onebar");
         bs=0;
      }
   }
}
if (bs==-1) {
   if (close<entry) {
      good+=1;
      if (good>=goodbars) {
```

```
            exitshort(0,close,"market","onebar");
            bs=0;
        }
    }
    else if (close>entry) {
        bad+=1;
        if (bad>=badbars) {
            exitshort(0,close,"market","onebar");
            bs=0;
        }
    }
}

if (mod(barnum,tradebars)==0) {
    rand=random(1000);
    if (rand>=500) {
        buy(1,close,"market","onebar");
        entry=close;
        good,bad=0;
        bs=1;
    }
    else {
        sell(1,close,"market","onebar");
        entry=close;
        good,bad=0;
        bs=-1;
    }
}
```

Earik_System_RandomTradeAll

Type: Indicator
Scaling: Screen
Reference Bars: 30
Description: This system trades every single bar, both to the long side and short side, and implements the random exit method to close trades. It requires the R11 function.

```
input: good(2), bad(6);

signal=0;
signal+=R11(good,bad);

if (barnum==barsback) {
   bs=0;
}

# reversal to upside
if ((signal>0) and (bs<=0)) {
   buy(signal,close,"market","onebar");
   bs=signal;
}

# reversal to downside
if ((signal<0) and (bs>=0)) {
   bs=signal;
   sell(-1*signal,close,"market","onebar");
}

# get flat
if (signal==0) {
   bs=0;
   exitlong(0,close,"market","onebar");
   exitshort(0,close,"market","onebar");
}

# adjust upside
if ((signal>0) and (bs>0) and (signal!=bs))  {
   if (signal>bs) {
      buy(signal-bs,close,"market","onebar");
      bs=signal;
```

```
    }
    if (signal<bs) {
        exitlong(bs-signal,close,"market","onebar");
        bs=signal;
    }
}

# adjust downside
if ((signal<0) and (bs<0) and (signal!=bs)) {
    if (signal<bs) {
        sell(abs(signal-bs),close,"market","onebar");
        bs=signal;
    }
    if (signal>bs) {
        exitshort(abs(signal-bs),close,"market","onebar");
        bs=signal;
    }
}

plot1=signal;
plot2=0;
```

R11

Type: Function

Reference Bars: 30

Description: This is a helper function that computes how many contracts long/short a system must be to follow the trade-every-bar random exit approach.

```
# this trades every single bar, both long or short, and exits
# using rexit method. Output is the net long/short position

input: nth_profitable_close, exit_nth_unprofitable_close;

define array bentry_price[];
define array sentry_price[];
define array bentry_for[];
define array bentry_against[];
define array sentry_for[];
define array sentry_against[];

if (barnum==barsback) {
   profit=0;
   position=0;
   good_exit=nth_profitable_close;
   bad_exit=exit_nth_unprofitable_close;
   bs=0;
}

# update all open longs
len=length(bentry_price);
for (i=0 to len-1 by 1) {
   entry=bentry_price[i];
   if (c>entry)
      bentry_for[i]+=1;
   if (c<entry)
      bentry_against[i]+=1;
}

# update all open shorts
len=length(sentry_price);
for (i=0 to len-1 by 1) {
   entry=sentry_price[i];
```

```
   if (c<entry)
      sentry_for[i]+=1;
   if (c>entry)
      sentry_against[i]+=1;
}

# update profits as necessary
len=length(bentry_price);
for (i=0 to len-1 by 1) {
   TOP1:
   if ((bentry_for[i]>=good_exit) or (bentry_against[i]>=bad_exit)) {
      profit+=close-bentry_price[i];
      remove(bentry_price,i);
      remove(bentry_for,i);
      remove(bentry_against,i);
      if (i==len-1) goto DONE1; else goto TOP1;
   }
}
DONE1:

len=length(sentry_price);
for (i=0 to len-1 by 1) {
   TOP2:
   if ((sentry_for[i]>=good_exit) or (sentry_against[i]>=bad_exit)) {
      profit+=sentry_price[i]-close;
      remove(sentry_price,i);
      remove(sentry_for,i);
      remove(sentry_against,i);
      if (i==len-1) goto DONE2; else goto TOP2;
   }
}
DONE2:

# compute net position
position = length(bentry_price) - length(sentry_price);

# we are buying and selling each bar, insert new trades
push(bentry_price,close);
push(bentry_for,0);
push(bentry_against,0);
```

```
push(sentry_price,close);
push(sentry_for,0);
push(sentry_against,0);

return position;
```

Earik_System_RandomOneContract

Type: Indicator
Scaling: Screen
Reference Bars: 30
Description: This trades the every-bar random approach, but only with one contract.

```
input: good(2), bad(6);

if (barnum==barsback) {
   bs=0;
}

signal=R11(good, bad);

# reversal to upside
if ((signal>0) and (bs<=0)) {
   buy(1,close,"market","onebar");
   bs=1;
}

# reversal to downside
if ((signal<0) and (bs>=0)) {
   bs=-1;
   sell(1,close,"market","onebar");
}

# get flat
if (signal==0) {
   bs=0;
   exitlong(0,close,"market","onebar");
   exitshort(0,close,"market","onebar");
}

plot1=signal;
plot2=0;
color2=blue;
style2=ps_dot;
```

Sign

Type: Function
Reference Bars: 3
Description: This is a helper function used in merging scripts. Simply returns whether the input number is positive or negative.

```
input:value;

if (barnum==barsback) {
   updn=0;
}

if (value>0) updn=1;
if (value<0) updn=-1;

return updn;
```

Earik_System_Merged

Type: Function
Reference Bars: 50
Description: This function merges 7 different systems together into a single timing line. Make sure to also enter the PUC_LunarSignal, hive_chicago, and sign functions, which are both used by this script. This function is used by the next four scripts.

```
# Need to compute each system and store it in a variable
if (barnum==barsback) {
   system1, system2, system3, system4, system5, system6, system7 = 0;
}

# Chicago Hive ------------------------------------------
system1 = hive_chicago();
system1 = sign(system1);

# GA1 ---------------------------------------------------
system2=0;

denom1 = close[1]-open[1];
denom2 = close-close[1];
if ((denom1!=0) and (denom2!=0)) {
   system2=(((close-high[1])/denom1) + (low[1]-low[2]))/denom2;
} else
   system2=0;
system2 -= ultramom(close,8,4);
system2 = sin(system2);

if ((low[2]-low[3]==1) or (high[2]-high[3]==1))
   system2 = close-high[1];

system2 = sign(system2);

# GA2 ---------------------------------------------------

usm2 = ultramom(close,2,4);
usm10 = ultramom(close,10,4);
usm18 = ultramom(close,18,4);

system3=0;
```

```
if (usm18!=0 and usm2!=0) {
   system3 = (1/usm18)*(1/usm18) - (1/usm2);
   sine_val = sin(usm10);
   if (system3 > sine_val) system3 = 1;
   else if (system3 < sine_val) system3 = -1;
}


# BCI (off the close!) ---------------------------------------

trigger=sqrt(h*l);
if (open<=trigger[1])
   system4 = 1;
else
   system4 = -1;


# Lunar Reversal ---------------------------------------------

system5=PUC_LunarSignal(112,5,175,true,true);

# UltraMom Inverse -------------------------------------------

mom=ultramom(c,2,4);
if (mom crosses over 0) system6 = -1;
if (mom crosses under 0) system6 = 1;

#Random System ---------------------------------------------

system7 = R11(3,10);
system7 = sign(system7);

# MERGED SYSTEM ----------------------------------------------
# we now have system1 through system7, each with a buy/sell value
# merge with simple voting

total = system1+system2+system3+system4+system5+system6+system7;

return total;
```

Earik_System_MergedVoting

Type: Indicator
Scaling: Screen
Reference Bars: 30
Description: This system trades the 7 merged signals. Make sure to include the Earik_System_Merged function.

```
# obtain the merged timing line
total = Earik_System_Merged();

if (barnum==barsback) {
   bs=0;
}

if ((total>0) and (bs<=0)) {
   buy(1,close,"market","onebar");
   bs=1;
}
if ((total<0) and (bs>=0)) {
   sell(1,close,"market","onebar");
   bs=-1;
}

plot1 = total;
plot2= 0;
```

Earik_System_SimplePosSizing

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: This system applies simple position sizing to the merged voting system.

```
input: initial_account(100000), one_contract_per(15000);

if (barnum==barsback) {
   bs=0;
   account=initial_account;
}

# obtain the merged timing line
total = Earik_System_Merged();

# update account
account = initial_account + (equitycurve+openequity)*50;

# figure out how many contracts to trade, and round down
cts = int(account/one_contract_per);

if ((total>0) and (bs<=0)) {
   buy(cts,close,"market","onebar");
   bs=1;
}
if ((total<0) and (bs>=0)) {
   sell(cts,close,"market","onebar");
   bs=-1;
}

plot1 = account;
```

Earik_System_ModifiedPosSizing

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: This system uses modified position sizing on the merged voting system.

```
input: initial_account(100000), one_contract_per(15000),
   rebalance_higher(0.05), rebalance_lower(0.3);

if (barnum==barsback) {
   bs=0;
   account=initial_account;

   #starting pos size
   cts=int(account/one_contract_per);
   if (cts<1) cts=1;
   rebalance_up=account+account*rebalance_higher;
   rebalance_dn=account-account*rebalance_lower;
}

# obtain the merged timing line
total = Earik_System_Merged();

# update account
account = initial_account + (equitycurve+openequity)*50;

# figure out contracts
if ((account crosses over rebalance_up) or (account crosses under rebalance_dn))
{
   cts=int(account/one_contract_per);
   if (cts<1) cts=1;
   rebalance_up=account+account*rebalance_higher;
   rebalance_dn=account-account*rebalance_lower;
}


if ((total>0) and (bs<=0)) {
   buy(cts,close,"market","onebar");
   bs=1;
}
```

```
if ((total<0) and (bs>=0)) {
   sell(cts,close,"market","onebar");
   bs=-1;
}

plot1 = account;
```

Earik_System_StockPosSize

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: This system uses position sizing on stocks, buying/selling as many contracts as possible given a certain account size and margin allowance

---

```
# this script will buy/sell as many shares as possible each trade
# margin_rate = 1 means no margin, 0.5 = 50% margin (range = 0.5 to 1)
input: initial_acct(10000), margin_rate(1);

if (barnum==barsback) {
   bs=0;
   shares=1;
   account=initial_acct;
   lastentry=close;
}

# obtain the merged timing line
total = Earik_System_Merged();

if ((total>0) and (bs<=0)) {
   if (bs<0)
      account+=(lastentry-close)*shares;
   shares=account/(close*margin_rate);
   buy(shares,close,"market","onebar");
   lastentry=close;
   bs=1;
}
if ((total<0) and (bs>=0)) {
   if (bs>0)
      account+=(close-lastentry)*shares;
   shares=account/(close*margin_rate);
   sell(shares,close,"market","onebar");
   lastentry=close;
   bs=-1;
}

plot1 = total;
plot2= 0;
```

Autotrade_Template

Type: Indicator
Scaling: Screen
Reference Bars: 50
Description: This is a template to show how to transform any system into an autotrading system that places trades hands-free. This script has been taken from one of my own systems which uses modified position sizing and Interactive Brokers. It stores the rebalancing values in a file, so they don't change between system loads. This file is accessed and updated by the indicator as position sizing moves along. The best place to put this file is in Wave59's working directory, which can be found by clicking File – Working Directory from within the program. You will also have to edit the margin and worstloss values according to your own system metrics. This script may contain bugs, and is only provided here for educational purposes to show how to go about doing something like this.

```
# this implements the XXXXX system for live trading with IB
# use the XXXXX function for actual signal generation
# requires daily chart with symbol = <enter your symbol here>
# implements Earik's Percentages Position Sizing, using text
#   files to store information between runs
# need to set the closing time to 3:14 rather than 3:15ct, will place
#   limit order at that price, with a go to market time at 3:14:40

# possizefile: location of file to save rebalancing values for pos sizing algorithm


input: possizefile("ENTER YOUR FILENAME HERE"), margin(5000), worstloss(8862),
worstloss_mult(1);

# initialization block ----------------------------------------------------
if (barnum==barsback) {

   # general purpose vars
   bs,signal,cts=0;
   define array posfile[];

   # pos sizing parameters
   rebalance_higher = 0.01;
   rebalance_lower = 0.3;  # ENTER YOUR OWN VALUE HERE
   increase_every = margin + worstloss * worstloss_mult;
```

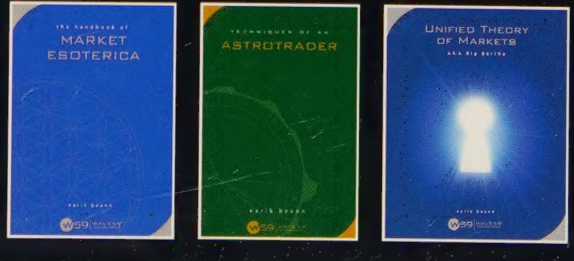
```
   # when to go to market on unfilled limits (set for central time)
   gomarket_hour = 15;
   gomarket_minute = 14;
   gomarket_second = 40;

   # error check to make sure chart is set to 1314 rather than 1315 endtime
   #if (endtime != 1314)
   #   Alert("Warning! Chart end time needs to be 1 minute before the close!","");
}

# figure out position sizing ---------------------------------------------
if (barnum>=updating_bar-1)
{
   acct_balance = IB_GetAccountValue("CashBalance");
   if (acct_balance == "") acct_balance = 0;
   acct_balance = to_double(acct_balance);

   # if pos size file doesn't exist, need to create it
   # pos size file contains three lines:
   #   rebalance_up level - if acct value passes, recompute
   #   rebalance_dn level - if acct value falls below, recompute
   #   contracts_to_trade - current num of contracts to trade
   if (file_exists(possizefile)==false) {
      cts= int(acct_balance/increase_every);
      if (cts<1) cts=1;
      rebalance_up=acct_balance+acct_balance*rebalance_higher;
      rebalance_dn=acct_balance-acct_balance*rebalance_lower;
      file_write(possizefile,rebalance_up);
      file_write(possizefile,rebalance_dn);
      file_write(possizefile,cts);
   }

   # read in the current position sizing levels and position size
   clear(posfile);
   file_read(posfile,possizefile);
   rebalance_up=to_double(posfile[0]);
   rebalance_dn=to_double(posfile[1]);
   cts=to_int(posfile[2]);
```

```
   # do we need to recompute these levels?
   if ((acct_balance > rebalance_up) or (acct_balance < rebalance_dn))
   {
      file_delete(possizefile);
      cts=int(acct_balance/increase_every);
      rebalance_up=acct_balance+acct_balance*rebalance_higher;
      rebalance_dn=acct_balance-acct_balance*rebalance_lower;
      file_write(possizefile,rebalance_up);
      file_write(possizefile,rebalance_dn);
      file_write(possizefile,cts);
   }
}

# trading system ------------------------------------------------------

signal = YOUR_SIGNAL_FUNCTION(); # you need your own trading signal here
position = ib_marketposition();  #how many long or short we already are

# longs
if ((signal>0) and (bs<=0)) {
   buy(1,close,"market","onebar");
   bs=1;

   # live orders
   if (position<=0) {
      contracts = cts + abs(position); # how many contracts to trade
      IB_CancelAll(); # cancel any pending orders, just in case!
      ref=IB_PlaceOrder("buy",contracts,close,"limit","day","");
      ib_gomarketattime(ref, year, month, day, gomarket_hour, gomarket_minute,
gomarket_second);
   }
}

# shorts
if ((signal<0) and (bs>=0)) {
   sell(1,close,"market","onebar");
   bs=-1;

   # live orders
   if (position>=0) {
```

```
      contracts = cts + abs(position); # how many contracts to trade
      IB_CancelAll(); # cancel any pending orders, just in case!
      ref=IB_PlaceOrder("sell",contracts,close,"limit","day","");
      ib_gomarketattime(ref, year, month, day, gomarket_hour, gomarket_minute,
gomarket_second);
   }
}

# plots ----------------------------------------------------------------
plot1=signal;
plot2=0;
color1=yellow;
```

Earik Beann manages several private funds and is developer of the Wave59® trading platform. He is the creator of a large number of proprietary market forecasting tools, and an avid researcher into the link between natural law and market movements. Earik has authored six trading books and numerous articles.

Also available at the Wave59 online bookstore:



Wave59.com/eshop

$1,995.00
ISBN 978-0-9910406-05
59999>

Wave59.com

WAVE59® TECHNOLOGIES